Review

Exploring The Model-View-Controller (MVC) Architecture: A Broad Analysis of Market and Technological Applications

Sabina Necula *

Posted Date: 29 April 2024

doi: 10.20944/preprints202404.1860.v1

Keywords: Model-View-Controller (MVC); Software Architecture; Web Development Frameworks; Enterprise Software Applications; AI Integration in Software; Machine Learning and MVC; Educational Tools in Software Engineering; Model-View-ViewModel (MVVM); JavaScript Frameworks; Agile Development

Review

Exploring the Model-View-Controller (MVC) Architecture: A Broad Analysis of Market and Technological Applications

SABINA-CRISTIANA NECULA

Accounting, Business Information Systems and Statistics Department, Faculty of Economics and Business Administration, Alexandru Ioan Cuza University of Iasi, Iasi, Romania; Sabina.necula@uaic.ro

Abstract: The Model-View-Controller (MVC) architecture, a cornerstone of modern software engineering, offers a robust framework for developing dynamic user interfaces. This paper conducts a broad analysis of MVC's deployment across various sectors, emphasizing its critical role in technological advancement and market integration. By examining implementation strategies in web development, mobile applications, and enterprise solutions, the study illustrates how MVC enhances application manageability, promotes efficient code reuse, and facilitates parallel development. The paper explores the adoption of MVC principles within major technologies such as Java, .NET, and JavaScript frameworks, underlining its widespread acceptance. Additionally, the analysis covers the preferential use of MVC frameworks within diverse market sectors including technology enterprises, educational institutions, e-commerce, and financial services, which leverage MVC for superior data management and user experience. Emerging trends such as the shift towards the Model-View-ViewModel (MVVM) architecture and the integration of MVC with artificial intelligence and machine learning technologies are also discussed. This comprehensive review not only reaffirms MVC's enduring relevance but also its adaptive evolution in response to emerging technological demands and market needs.

Keywords: Model-View-Controller (MVC); software architecture; web development frameworks; enterprise software applications; AI integration in software; machine learning and MVC; educational tools in software engineering; Model-View-ViewModel (MVVM); JavaScript frameworks; agile development

JEL Classification: O31; O33; L86; C88

1. Introduction

The Model-View-Controller (MVC) architecture is a seminal design pattern that has profoundly influenced the landscape of software development since its inception[1]. Originally conceptualized for desktop graphical user interfaces, MVC has evolved to become a cornerstone framework guiding the development of web and mobile applications. Its ability to separate concerns—dividing the application logic into three interconnected components: the model, the view, and the controller—facilitates not only enhanced code reuse but also parallel development across teams [2,3]. This architectural pattern enables developers to manage complex applications efficiently, making it a popular choice among a wide array of industries.

Given its essential role, a thorough analysis of MVC's applications across different technologies and markets is essential. This study aims to explore the breadth and depth of MVC's application, highlighting how different sectors leverage this pattern to optimize software development and address the unique challenges posed by their respective fields. From web frameworks that implement MVC in various programming languages to mobile platforms that depend on it for application structure, MVC's versatility is showcased across diverse technological landscapes. When examining the market and technologies where the This design pattern separates the application into three interconnected components, allowing for efficient code reuse and parallel development.

Technologies Using MVC are: 1) Web Development: MVC is widely adopted in web development frameworks across various programming languages. Key technologies include Java (frameworks like Spring MVC allow for the creation of dynamic and robust web applications) [4–7], .NET (ASP.NET MVC framework is used extensively for building enterprise-level web applications) [8,9], and JavaScript (Frameworks like Angular, React, and Vue.js use variations of MVC principles to enhance front-end development)[10]; 2. Mobile Applications: MVC is employed in the development of mobile applications to manage complex user interfaces and data interactions, particularly in iOS Development. Apple's iOS frameworks use MVC extensively to structure applications in a way that isolates the user interface from the business logic[11,12]; 3. Enterprise Applications: MVC supports the development of scalable enterprise applications, ensuring that large systems are well-organized in terms of code manageability and database handling.

The primary market for MVC architecture is the technology sector, especially companies focusing on software development and web application. This encompasses everything from startups to large enterprises needing robust application architectures. Moreover, as MVC is a fundamental architectural pattern, educational institutions [13,14] and training programs include it extensively in computer science and software engineering courses.

Another market area is represented by E-Commerce. Many e-commerce platforms utilize MVC frameworks to handle the vast and dynamic content inherent in online shopping systems, helping manage user interactions, backend processing, and data integrity [15–18]. A preeminent market area is represented by the financial services industry [19]. MVC frameworks are used to develop applications that require rigorous data integrity, security, and user interface flexibility, characteristics needed in banking and finance software.

This paper also addresses the evolving nature of MVC, considering emerging trends that influence its implementation and effectiveness, such as the rise of Model-View-ViewModel (MVVM) architectures and the integration of advanced computational technologies like artificial intelligence and machine learning.

2. Methodology for Systematic Literature Review on MVC Architecture

This study employs a systematic literature review aimed at exploring the applications of the Model-View-Controller (MVC) architecture within various software architectures and design patterns, as well as understanding the markets and technologies where MVC is most prevalently used. Our study is guided by the following research questions:

RQ1- Implementation Variability: How is MVC implemented across different programming languages and platforms?

RQ2- Architectural Impact: What impacts does MVC have on software architecture and design patterns within these contexts?

RQ3- Market and Technological Alignment: Which markets and technologies most frequently employ MVC?

To ensure a thorough and relevant collection of data, our literature search encompasses a wide range of prominent academic databases including Scopus, Clarivate Analytics (Web of Science), SpringerLink, and the ACM Digital Library. The search strategy employs a combination of targeted keywords to capture the broad spectrum of MVC applications. These keywords include:

- Primary Keywords: "Model-View-Controller" or "MVC"
- Contextual Keywords: "software architecture" or "design patterns"
- Technological Keywords: Java, C#, JavaScript, Python, or Ruby

These keywords are strategically combined to maximize the retrieval of pertinent studies into the following search phrase: ("Model-View-Controller" OR "MVC") AND ("software architecture" OR "design patterns") AND (java OR "C#" OR "JavaScript" OR python OR ruby). We used this search phrase over academic databases.

The inclusion criteria are strictly defined to consider only peer-reviewed articles that explicitly discuss the application or theoretical analysis of MVC, written in English. Publications that do not directly address MVC, are not peer-reviewed, or are irrelevant to the predefined programming

doi:10.20944/preprints202404.1860.v1

languages are excluded. From each selected publication, key information is extracted, including bibliographic details, study focus, key findings, and methodologies.

The quality of each study is rigorously assessed to ensure high standards of research integrity and relevance. This assessment considers factors such as the clarity of research objectives, the appropriateness of research methods, and the robustness of conclusions. After data extraction, we synthesize the findings to identify common themes, trends, and research gaps related to the technological and market applications of MVC.

To further enhance our analysis, we employed the Bibliometrix R package and its web interface, Biblioshiny, to perform a bibliometric analysis of the collected data. This approach allowed us for a deeper quantitative analysis of the literature, helping to map the intellectual structure of MVC research and its trajectory over time. Specifically, we used:

1. Co-occurrence Analysis: Identify the most frequently cited works and terms within the MVC literature to discern key theories and methodologies.
2. Collaboration Networks: Analyze collaboration patterns among authors, institutions, and countries to understand the geographic and institutional distribution of MVC research.
3. Trend Analysis: Use the annual scientific production and citation trends to gauge the growth and impact of MVC-related research.
4. Thematic Map Analysis: Explore the strategic diagram created by Bibliometrix to classify themes into clusters based on their centrality and density, offering insights into well-developed areas and emerging trends.

The synthesized data from both the systematic literature review and the bibliometric analysis are compiled into a comprehensive report. This report outlines the prevalent use of MVC in software development, delineates key trends, discusses the implications of the findings, and suggests potential directions for future research. This systematic and analytical approach provides a thorough exploration of the MVC architecture, offering significant insights into its application and implications across diverse software development environments and market sectors.

3. Results

Table 1 presents the bibliometric overview of MVC research. The extracted 123 documents indicate a body of research that is large enough to represent the field, with an annual growth rate of 7.47%, suggesting a healthy and growing interest in MVC research. An average document age of 11.6 years, coupled with an average of 4.236 citations per document, underscores the lasting relevance and the impact of the research in this area over time. The involvement of 347 authors, with a moderate level of international co-authorship at 12.2%, reflects a collaborative and global research community. An average of 3.05 co-authors per document indicates teamwork in conducting MVC studies, which is typical for interdisciplinary research fields like software architecture. The predominance of conference papers (75 out of 123 documents) indicates that MVC is a dynamic field with ongoing discussions and developments primarily shared in conferences. The presence of articles, books, book chapters, and reviews highlights the topic's maturity and depth, with substantial contributions that warrant book-level treatment and peer-reviewed articles that contribute to the body of knowledge in a significant way.

Table 1. Bibliometric Overview of MVC Architecture Research (1996-2023).

Description	**Results**
MAIN INFORMATION ABOUT DATA	
Timespan	1996:2023
Sources (Journals, Books, etc)	99
Documents	123
Annual Growth Rate %	7.47
Document Average Age	11.6
Average citations per doc	4.236

References	1
DOCUMENT CONTENTS	
Keywords Plus (ID)	1096
Author's Keywords (DE)	369
AUTHORS	
Authors	347
Authors of single-authored docs	22
AUTHORS COLLABORATION	
Single-authored docs	25
Co-Authors per Doc	3.05
International co-authorships %	12.2
DOCUMENT TYPES	
article	34
article conference paper	1
book	5
book chapter	3
conference paper	75
conference review	3
review	2

3.1. Co-Occurrence Analysis

Figure 1 presents the co-occurrence of the keywords used by the analyzed articles.

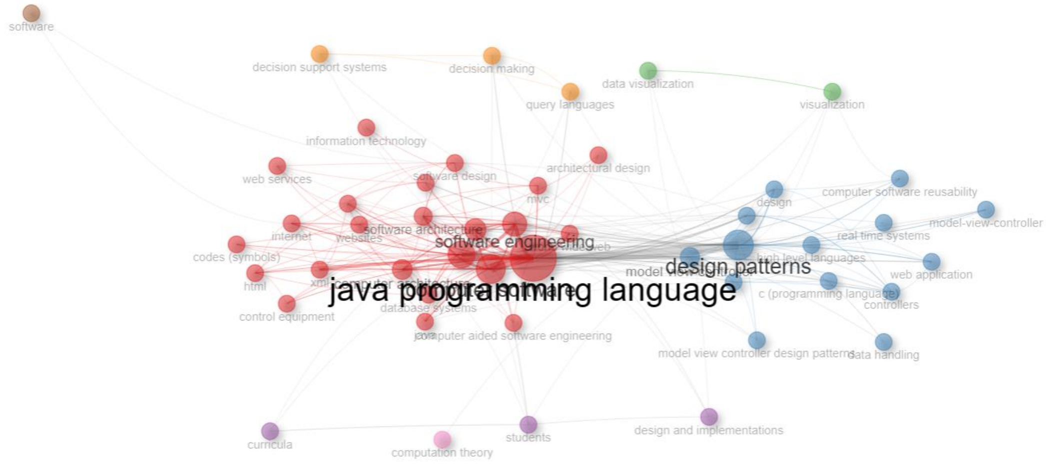

Figure 1. Co-occurrence network.

"Java programming language" appears as a highly central node, indicating its important role in discussions related to MVC. This suggests that a significant portion of MVC-related research involves Java, likely due to its widespread use in web and enterprise applications. "Design patterns" is also of high centrality, specifically in its own cluster, indicating a strong focus on MVC within the context of software design best practices.

Cluster 1 seems to focus on general technological terms associated with MVC, such as "user interfaces" and "software engineering." These keywords are central to MVC discussions, possibly addressing the practical applications of MVC.

Cluster 2 includes terms like "model view controller design patterns," suggesting a specialized focus on how MVC is implemented within the broader context of design patterns.

doi:10.20944/preprints202404.1860.v1

Keywords with high betweenness but belonging to different clusters can be indicative of interdisciplinary research areas that bridge MVC with other fields, such as "design patterns" and "model view controller design patterns."

Terms like "computation theory" and "software" with lower centrality measures may represent more niche or emerging areas of MVC research, potentially indicating new directions or less explored applications of MVC architecture.

"Students" and "decision making" in separate clusters indicate specific contexts in which MVC research is applied, such as educational tools or decision support systems.

The occurrence of "data visualization" and "visualization" in their own cluster suggests a specialized niche within MVC research focused on how MVC frameworks can be utilized for visualizing data.

3.2. Collaboration Network

The collaboration network (Figure 2, Table 2) highlights the frequency of joint research efforts between countries in the field of Model-View-Controller (MVC) architecture.

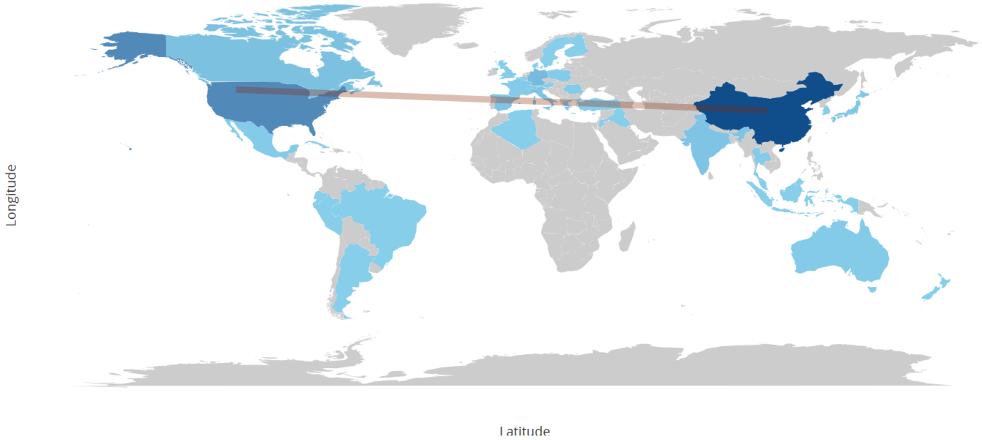

Figure 2. Collaboration network.

Table 2. Collaboration network data.

From	To	Frequency
Australia	Romania	1
Canada	Denmark	1
Canada	Greece	1
Canada	Turkey	1
China	Australia	1
China	France	1
China	Japan	1
China	Usa	2
Denmark	Turkey	1
Finland	United Kingdom	1
France	Algeria	1
Germany	Cuba	1
Spain	Ecuador	1
Spain	Japan	1
Usa	Argentina	1
Usa	Spain	1

The collaborations are predominantly unique pairings between countries, as indicated by a frequency of 1. This suggests that while international collaborations do occur in MVC research, extensive or recurring partnerships are not as common, possibly pointing to opportunities for more sustained international research relationships. Canada shows diverse research partnerships with Denmark, Greece, and Turkey, indicating its engagement in international collaborations. China also displays multiple collaborations with countries like Australia, France, Japan, and notably, with the USA (with a frequency of 2), highlighting its active participation in global MVC research. The collaborations span across continents, with connections such as Australia and Romania or the USA and Argentina. This denotes the worldwide interest in MVC architecture research and its cross-border relevance. Countries like the USA, China, and Spain, which appear multiple times as nodes in these collaborations, may be viewed as potential hubs for MVC research, attracting partnerships with various other nations. Single instances of collaboration, such as Germany with Cuba or Spain with Ecuador, could indicate specific research projects or agreements that brought these nations together for MVC-related studies.

3.3. Trend Analysis

Figure 3 presents the trend topics of research in this field. Topics like "java programming language," "user interfaces," and "software engineering" have a high frequency of publications spread over an extended period, indicating sustained and possibly mature lines of research.

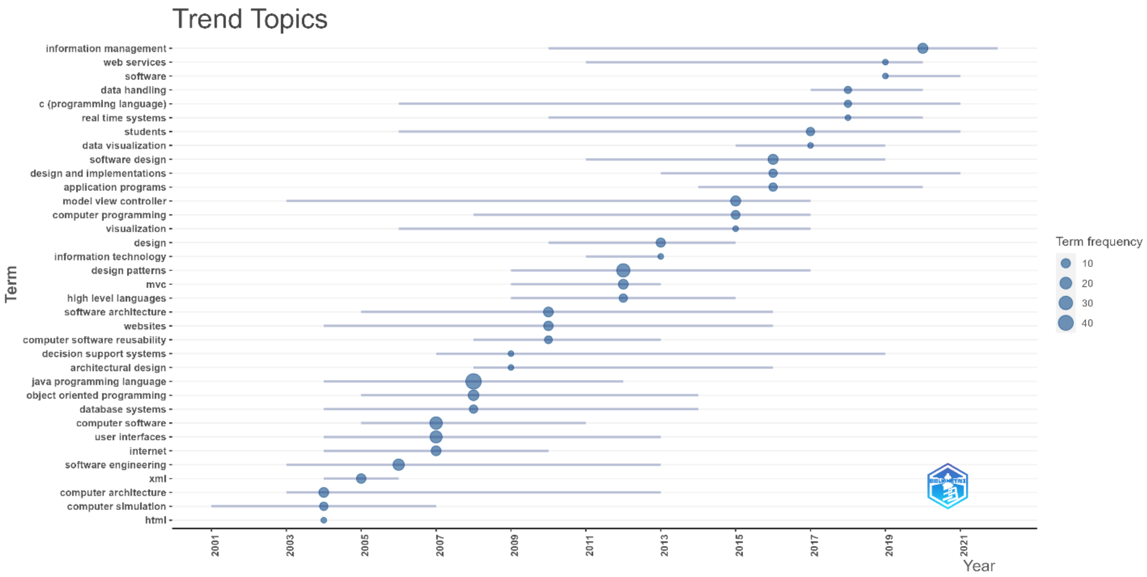

Figure 3. Trend topics.

- "Information management" and "software design" show a later median year (2016-2020), suggesting these areas have gained or regained significant attention in recent years.
- "HTML" has all its quartiles within 2004, suggesting a concentrated spike in research, possibly due to specific technological developments or standards being released in that year.
- "Computer architecture" and "computer simulation" have their third quartile earlier than 2013, which might indicate a plateau or decline in new research in these areas.
- Topics such as "real time systems," "data handling," and "web services" have their third quartile post-2018, indicating these are current research interests that may continue to grow.
- The "java programming language" and "object-oriented programming" show a broad span from early to later quartiles, reflecting long-standing research interest and development within these areas.
- "MVC" itself shows a focused peak between 2009 and 2013, which may coincide with the rise of web frameworks and mobile application development emphasizing the MVC pattern.
- "Information management" demonstrates recent and quick growth with a median year of 2020, possibly due to digital transformation trends and big data.

The thematic evolution is depicted in Figure 4.

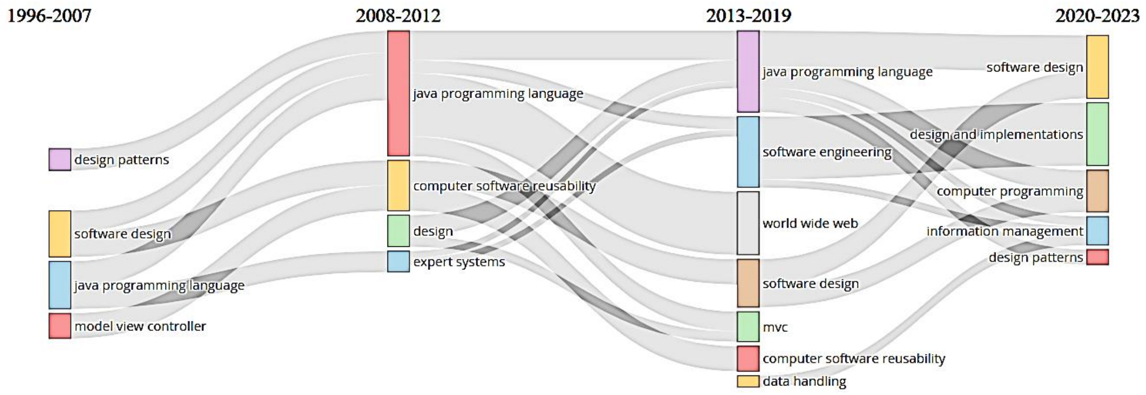

Figure 4. Thematic evolution.

To characterize the shifts between the main themes we present in Table 3 the metrics for the thematic evolution.

Table 3. Thematic evolution metrics.

From	To	Weighted Inclusion Index	Inclusion Index
java programming language--2007-2012	world wide web--2013-2019	1.00	1.00
software engineering--2013-2019	design and implementations--2020-2023	1.00	1.00
java programming language--2013-2019	software design--2020-2023	0.57	0.33
computer programming--1996-2006	java programming language--2007-2012	0.50	0.50
computer software reusability--1996-2006	java programming language--2007-2012	0.50	0.50
java programming language--2007-2012	java programming language--2013-2019	0.47	0.06
java programming language--1996-2006	java programming language--2007-2012	0.46	0.04
software design--2013-2019	software design--2020-2023	0.43	0.33
design--2007-2012	java programming language--2013-2019	0.33	0.20
java programming language--2007-2012	computer software reusability--2013-2019	0.33	0.33
java programming language--2007-2012	software engineering--2013-2019	0.33	0.11
java programming language--2013-2019	computer programming--2020-2023	0.33	0.33
software design--2013-2019	computer programming--2020-2023	0.33	0.33
java programming language--2007-2012	mvc--2013-2019	0.31	0.17
c (programming language)--1996-2006	markup languages--2007-2012	0.25	0.25

model view controller--1996-2006	java programming language--2007-2012	0.25	0.25
java programming language--2013-2019	design patterns--2020-2023	0.24	0.13
java programming language--2007-2012	software design--2013-2019	0.21	0.25
data handling--2013-2019	information management--2020-2023	0.18	0.20
design--2007-2012	mvc--2013-2019	0.17	0.20
java programming language--2013-2019	information management--2020-2023	0.15	0.06

Java Programming Language and E-Commerce to Database Systems (2007-2012) evolution explains:

- The connection between Java, often used in backend development, and its application in e-commerce systems to manage database systems.
- The low weighted and inclusion indexes but higher occurrence suggest a less dominant but consistently noted theme.

Java Programming Language and MVC (2007-2012 to 2013-2019) evolution explains:

- The evolution of Java programming towards integrating and using the Model-View-Controller (MVC) framework.
- A moderately high WII and II indicate a significant thematic shift and connection.

Software Engineering to Design and Implementations (2013-2019 to 2020-2023) evolution explains a very strong connection (both indexes at 1.00), suggesting a direct and significant evolution from general software engineering practices to specific design and implementation techniques.

There are multiple connections highlighting Java's evolving role in various contexts like design patterns, software design, and its continuous influence over time.

The varying indices reflect the differing levels of thematic strength and stability in each period.

3.4. Thematic Map

In order to visually interpret the complex landscape of MVC application and its interaction with emerging technologies, this section introduces a thematic map analysis (Figure 5). Thematic maps are powerful tools in bibliometric studies, enabling researchers to categorize and display the relationships between themes or topics within a large body of literature. This approach helps in identifying the centrality and density of each theme, providing insights into their influence and development within the field.

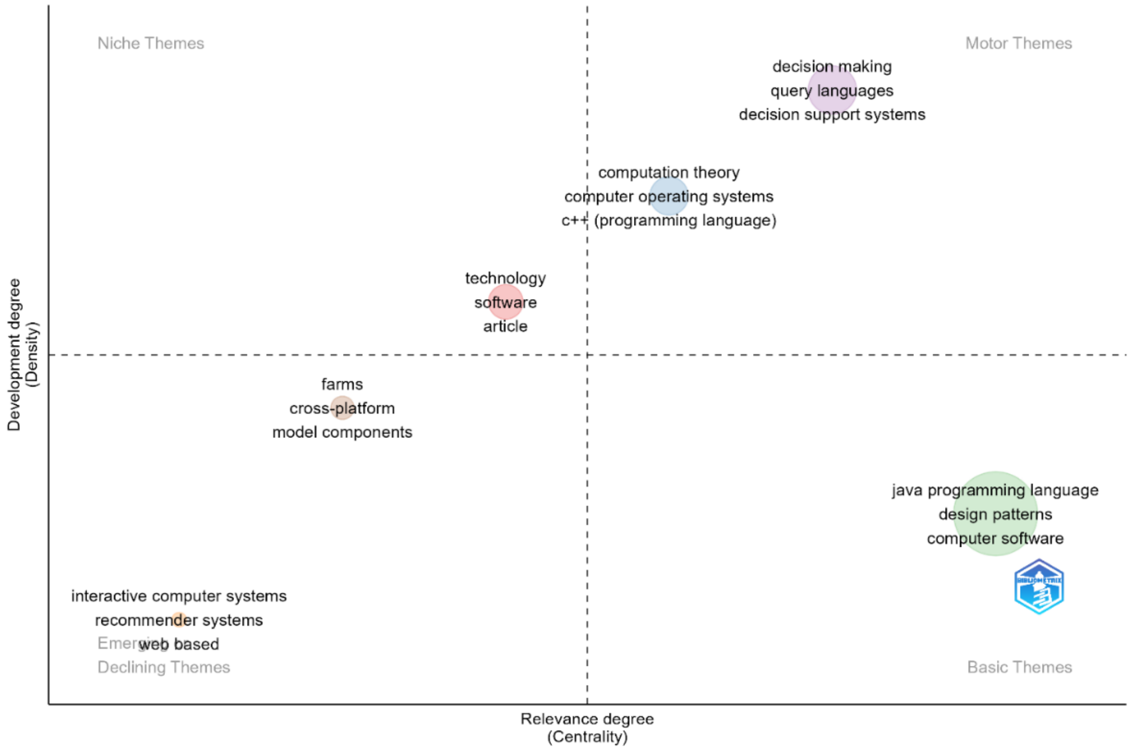

Figure 5. Thematic map.

This thematic map presents a snapshot of the research landscape within various areas related to Model-View-Controller (MVC) architecture and associated technologies. Each term is classified under different clusters that represent broad thematic categories. The centrality measures suggest the influence and importance of each term within the research network.

Cluster 1: Technology

Terms like "software," "technology," "article," and "management" have high betweenness centrality scores, which suggests they act as bridges connecting different sub-topics within the technology cluster. They may reflect key areas of focus and are often terms that appear in various contexts, signifying their broad relevance.[20,21] The presence of terms like "business logic" and "database management systems" indicate the applications of MVC in creating business-oriented software solutions that prioritize efficient data management.

Cluster 2: Computation Theory

Terms like "computation theory"[22,23], "computer operating systems" [24] and "C++ (programming language)"[25] or "C#"[26] signal in-depth, foundational research within computation, which underpins the development and use of MVC architectures. The presence of "computer control systems" and "database applications" highlights the theory's practical applications, particularly how MVC is implemented at the systems level and in managing databases within software applications.

Cluster 3: Java Programming Language

With high occurrences and centrality measures, "java programming language" is a central node, indicating its significant role in MVC development and research. This suggests a heavy use of Java in MVC frameworks, confirming its market and technological relevance. Terms like "design patterns"[14,16,20,21,25–28], "computer software" [36], "user interfaces" [37] and "software engineering" [22–24], indicate a strong emphasis on using MVC in software design and user interface development, with a focus on object-oriented programming for information management and software design. "Model view controller," "MVC," and "software architecture" are directly related to MVC and reflect the research dedicated to exploring MVC's application in software architectural design.

Implications for MVC Market and Technologies

The high frequency and centrality of "java programming language" and "software engineering" point to a vibrant market for MVC-based software development, with Java being a preferred language. The overlap between "design patterns" and "MVC" suggests that MVC remains a popular design pattern within software engineering education and practice, implying a steady demand in the technology job market. The inclusion of "computer software," "user interfaces," and "information management" indicates a wide range of technologies and domains that MVC is addressing, from front-end user interface frameworks to back-end information management systems.

3.5. Factorial Analysis

Factorial analysis (Figure 6) is often used in bibliometrics to identify and understand underlying patterns in data sets, especially to explore relationships among variables (in this case, likely keywords or topics) across dimensions. Dim.1 and Dim.2 indicate the scores of each theme on the first and second principal dimensions, respectively. These dimensions are derived to explain the maximum variance in the dataset.

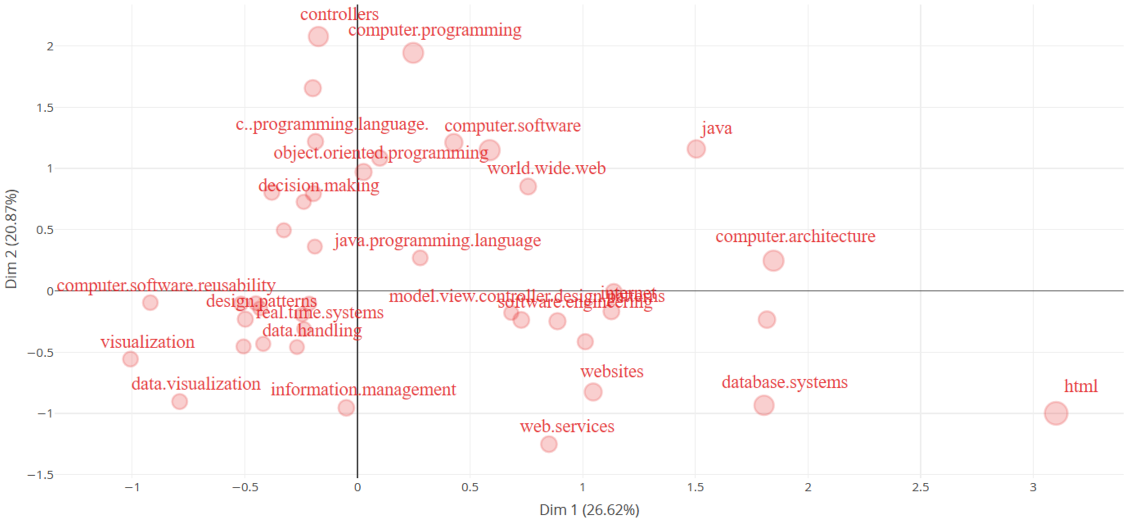

Figure 6. Factorial analysis.

Dimension 1 (Dim.1): This dimension might represent a specific aspect of technology or methodological approach within the field. Higher positive scores on this dimension suggest that themes are more strongly related or more significant concerning this aspect. Conversely, negative scores might indicate lesser relevance or opposite characteristics.

Dimension 2 (Dim.2): Similarly, this dimension represents another aspect or characteristic. The positioning of themes along this axis helps to differentiate them based on another set of criteria, which could be orthogonal (independently varying) to those captured by the first dimension.

Analysis of Specific Themes

- HTML (3.16, -0.87): Stands out with a very high score on Dim.1, suggesting it has a strong presence or influence concerning the factor captured by this dimension, perhaps indicating its foundational role in web technologies.
- Control Equipment (2.64, 0.27) and Database Systems (1.87, -0.83): These also have high scores on Dim.1, implying significant roles in their respective areas, possibly highlighting their technological importance or foundational aspects in system design and management.
- Web Services (0.4, -2.2): Shows an extreme score on Dim.2, indicating a unique or specialized role distinct from other themes along the second dimension, perhaps pointing to its specific applications or technological nuances.
- Software Design Patterns (-0.31, -1.95): This negative extreme on Dim.2 could suggest a divergence or specialized niche in design approaches compared to more general software engineering themes.

4. Discussions

The study emphasizes the MVC design pattern's role in creating scalable and maintainable applications. MVC is highlighted for its ability to separate concerns, which simplifies management and updates, crucial for adapting to changing requirements over time. Java is commonly chosen for its robustness and cross-platform capabilities, which are essential for enterprise applications.

Adoption of MVC in Diverse Applications

The works of Bodhuin et al. (2002) [27], Singh et al. (2018)[38], and Sauter et al. (2005) [39,40] illustrate the widespread adoption and versatility of the MVC pattern in transforming legacy systems, web development, and pervasive multi-client user interfaces. These studies suggest a continuing trend towards using MVC architectures due to their ability to separate concerns, which simplifies development and maintenance, enhancing scalability and flexibility in various application domains.

Integration with Modern Technologies

Zhao et al. (2010) [16] and Ding et al. (2012) [32] discuss the integration of MVC with modern frameworks like Struts2, Spring, and Hibernate. This trend points to MVC's adaptability and its role in facilitating the development of robust, scalable applications. Incorporating MVC with these technologies supports the construction of more dynamic, efficient, and manageable web applications, indicating a market trend towards frameworks that can seamlessly integrate with other advanced technological stacks.

Educational Tools and Frameworks

The development of educational platforms using MVC, as discussed by Ishihara et al. (2017) [13] and Jiang et al. (2023) [41], showcases the framework’s utility in creating educational software. This reflects a trend towards employing MVC architectures in the educational sector to develop applications that are both scalable and maintainable, thereby enhancing learning and teaching experiences through technology.

Enhancements in MVC for Specific Needs

Furtado et al. (2015) [42] and Sauter et al. (2004) [39] highlight modifications and extensions to the standard MVC pattern to meet specific industry needs such as information visualization in three-dimensional environments and pervasive computing. This trend indicates that as different sectors evolve, there is a growing need to adapt and extend traditional software architectures like MVC to meet specific performance and functional requirements.

Future Trends in MVC Applications:

The earlier discussions by Taraghi and Ebner (2010) [43] on a simple MVC framework for widget development and by Caballe et al. (2014) [44] on simplifying the development of Java EE application clients suggest a forward-moving trend towards simplifying MVC frameworks for specific functionalities. This indicates a market trend towards frameworks that are not only robust but also easier to use and implement, making them accessible to a broader range of developers.

These insights demonstrate a robust trend in the continued evolution and adaptation of the MVC framework across various fields and technologies. This trend is driven by the need for scalable, maintainable, and efficient software solutions that cater to specific industrial, educational, and technological needs.

Shift towards MVVM and MERN: While MVC remains foundational, there is a noticeable shift towards other patterns like Model-View-ViewModel (MVVM) and stacks like MERN (MongoDB, Express, React, Node.js), which reflect the evolving demands of real-time user interfaces and single-page applications.[10]

Integration with AI and Machine Learning: Incorporating AI functionalities into MVC architectures is becoming a trend, enhancing the capabilities of applications in terms of personalized user experiences and intelligent data processing. This integration primarily focuses on enhancing the capabilities of the Model component of MVC, where AI algorithms can be employed to improve data handling, processing, and decision-making processes based on user interactions captured through the Controller. Microsoft's ASP.NET Core MVC framework now includes features that facilitate the integration of ML.NET, a machine learning framework. The popular Java framework, Spring Boot, used with MVC design, can be integrated with TensorFlow, an open-source machine learning library. In Ruby on Rails, which follows the MVC pattern, developers can integrate Python's SciKit-Learn machine learning library using tools like Jupyter Notebook via APIs.

5. Conclusions

Given these insights, MVC continues to be an influential and versatile pattern in software architecture, adaptable across different technologies and markets, and remains relevant in the design of modern software applications.

Key Findings of our study

Versatility and Adoption

The MVC architecture's adaptability across various platforms and languages (Java, .NET, JavaScript) underscores its foundational role in modern software development. The studies reviewed indicate a widespread adoption of MVC frameworks not only in web and enterprise application development but also in mobile platforms, especially iOS, due to its effectiveness in managing complex data-driven interfaces.

Technological Integration

MVC's integration with modern development frameworks such as Spring MVC, .NET, and front-end JavaScript frameworks (Angular, React, Vue.js) illustrates its critical role in contemporary web and software development ecosystems. This integration facilitates the development of more dynamic, scalable, and efficient applications, catering to the needs of fast-paced, data-intensive market environments.

Market Dynamics

The MVC framework is particularly favored in technology-driven industries, including digital commerce and financial services, where robust, scalable, and secure applications are crucial. Its use in educational tools and platforms also highlights its role in sectors that require reliable and maintainable software architectures.

Evolution and Trends

The progression towards more component-based architectures like Model-View-ViewModel (MVVM) and the integration of MVC with emerging technologies such as artificial intelligence and machine learning indicate a shift towards more dynamic and user-centric applications. This evolution is driven by the need for applications to be more adaptive, intuitive, and capable of handling complex datasets with ease.

The enduring relevance of the MVC architecture in the face of evolving technological demands and market needs is a testament to its foundational principles. As industries continue to embrace digital transformation, MVC's principles of separation of concerns, modularity, and scalability make it an essential pattern in the toolbox of modern software developers. Future research could further explore the integration of MVC with AI and real-time data processing technologies to understand its potential in enhancing user experience and operational efficiency in software applications.

The insights derived from this comprehensive analysis not only affirm MVC's adaptability and enduring relevance but also highlight its evolving trajectory in response to new technological advancements and market demands. As such, MVC remains a cornerstone in the architecture of modern software systems, continually adapting to meet the challenges of the digital age.

doi:10.20944/preprints202404.1860.v1

References

1. Kim, Da-Jeong and Lee, Eun-Ser, "The VMDC(View, Model, Dispatcher, Controller) Architecture for Products Management," *The KIPS Transactions:PartD*, vol. 16D, no. 6, pp. 881–888, Dec. 2009, doi: 10.3745/KIPSTD.2009.16D.6.881.
2. M. Kalelkar, P. Churi, and D. Kalelkar, "Implementation of Model-View-Controller Architecture Pattern for Business Intelligence Architecture," *IJCA*, vol. 102, no. 12, pp. 16–21, Sep. 2014, doi: 10.5120/17867-8786.
3. S. Kim and K. Choi, "Comparing Model, View, and Controller Architecture and Logical User Interface Model," *adv sci lett*, vol. 22, no. 11, pp. 3274–3278, Nov. 2016, doi: 10.1166/asl.2016.7865.
4. C. Vichido, M. Estrada, and A. Sanchez, "A constructivist educational tool: Software architecture for Web-based video games," in *Proc. Mex. Int. Conf. Comp. Sci.*, Chavez E., Favela J., Oliart A., and Mejia M., Eds., IEEE Computer Society, 2003, pp. 144–150. doi: 10.1109/ENC.2003.1232888.
5. M. Hillenbrand and B. Reuther, "Building blocks for Web applications," in *Proc. IASTED Internat. Conf. Internet Multimed. Sys. Appl.*, 2003, pp. 757–761. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-1542642675&partnerID=40&md5=bf7271c1e78ebec902a5d4a3b314fca1
6. N. F. Polys, F. Bacim, M. Setareh, and B. Jones, "Building novel Web3D user interfaces - A case study from architecture and structural engineering," in *Proc. - Web3D: Int. Conf. 3D Web Technol.*, 2013, pp. 135–138. doi: 10.1145/2466533.2466554.
7. R. DuPlain, J. Benson, and E. Sessoms, "Data vault: providing simple web access to NRAO data archives," in *Proc SPIE Int Soc Opt Eng*, 2008. doi: 10.1117/12.789402.
8. J. D. Alanís Urquieta, G. Sánchez Flores, M. P. Morales Riveroll, E. López Mendez, and B. J. Blanca, "Experimental comparison of the implementation of MVC in Java and C#," in *WMSCI - World Multi-Conf. Syst., Cybern. Informatics, Proc.*, Callaos N.C., Savoie M., Tremante A., and Sanchez B., Eds., International Institute of Informatics and Systemics, IIIS, 2017, pp. 192–197. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85032358022&partnerID=40&md5=9f794fd859812101194bdbd11b3234b1
9. F. A. Masoud, D. H. Halabi, and D. H. Halabi, "ASP.NET and JSP frameworks in model view controller implementation," in *Proc. - Int. Conf. Inf. Commun. Technol.: Theory Appl., ICTTA*, Institute of Electrical and Electronics Engineers Inc., 2006, pp. 3593–3598. doi: 10.1109/ICTTA.2006.1684998.
10. R. Steyer, *Building web applications with Vue.js: MVVM patterns for conventional and single-page websites*. in Building web applications with Vue.js: MVVM patterns for conventional and single-page websites. Springer, 2022, p. 219. doi: 10.1007/978-3-658-37596-6.
11. J. Narsoo, M. S. Sunhaloo, and R. Thomas, "The application of design patterns to develop games for mobile devices using java 2 micro edition," *J. Object Technol.*, vol. 8, no. 5, pp. 153–175, 2009, doi: 10.5381/jot.2009.8.5.a4.
12. F. Hengming, C. Jia, and X. Bin, "The interaction mechanism based on JSON for android database application," *Inf. Technol. J.*, vol. 12, no. 1, pp. 224–228, 2013, doi: 10.3923/itj.2013.224.228.
13. N. Ishihara, N. Funabiki, M. Kuribayashi, and W.-C. Kao, "A proposal of software architecture for Java programming learning assistant system," in *Proc. Int. Conf. Adv. Inf. Netw. Appl. AINA*, Enokido T., Hsu H.-H., Lin C.-Y., Takizawa M., and Barolli L., Eds., Institute of Electrical and Electronics Engineers Inc., 2017, pp. 64–70. doi: 10.1109/AINA.2017.56.
14. N. Dehbozorgi and A. Norkham, "An Architecture Model of Recommender System for Pedagogical Design Patterns," in *Proc. Front. Educ. Conf. FIE*, Institute of Electrical and Electronics Engineers Inc., 2021. doi: 10.1109/FIE49875.2021.9637342.
15. W. F. Zhang, "Design and implementation workflow management system based on JAVA technology," in *Proc. Intern. Conf. Inf. Technol. Comput. Appl. Eng. ITCAE*, Liu H.-C., Sung W.-P., and Yao W., Eds., CRC Press/Balkema, 2015, pp. 537–540. doi: 10.1201/b18658-125.
16. C. Zhao, M. Jiang, and Z. He, "The design of E-commerce system architecture based on Struts2, Spring and Hibernate," in *Int. Conf. Inf. Sci. Eng., ICISE - Proc.*, 2010, pp. 3251–3254. doi: 10.1109/ICISE.2010.5690571.
17. M. Mouratidou, V. Lourdas, A. Chatzigeorgiou, and C. K. Georgiadis, "An assessment of design patterns' influence on a java-based e-commerce application," *J. Theor. Appl. Electron. Commer. Res.*, vol. 5, no. 1, pp. 25–38, 2010, doi: 10.4067/S0718-18762010000100004.

18. F. Li and J. Zhang, "Construction and Realization of the Marketing Management Information System for E-commerce Companies Based on SQL Server," in *ACM Int. Conf. Proc. Ser.*, Association for Computing Machinery, 2021, pp. 389–393. doi: 10.1145/3510858.3510972.
19. M. A. Arias-Serna, E. P. Luna-Moreno, J. H. Hoyos-Barrios, and J. A. Echeverri-Arias, "Development of an application for the calculation of credit risk in entities of the Colombian solidarity sector," *Rev. Iberica Sist. Tecnol. Inf.*, vol. 2022, no. E52, pp. 174–188, 2022.
20. N. Wang and F. Xiao, "Design of logistics information management system based on database," in *ICCMS - Int. Conf. Comput. Model. Simul.*, 2010, pp. 111–115. doi: 10.1109/ICCMS.2010.343.
21. A. Umer *et al.*, "A decision support system for diagnostics and treatment planning in traumatic brain injury," *IEEE J. Biomedical Health Informat.*, vol. 23, no. 3, pp. 1261–1268, 2019, doi: 10.1109/JBHI.2018.2842717.
22. V. V. Savinykh, A. N. Borovski, and O. V. Postylyakov, "Model of development of cross-platform software for ozone and ultraviolet radiation measurements on the example of Brewer Spectrophotometer," in *Proc SPIE Int Soc Opt Eng*, Matvienko G.G. and Romanovskii O.A., Eds., SPIE, 2019. doi: 10.1117/12.2540656.
23. V. V. Savinykh and O. V. Postylyakov, "On the development of a cross-platform database application for storing long-term observations of ultraviolet radiation and total ozone content obtained using Brewer Spectrophotometer," in *Proc SPIE Int Soc Opt Eng*, Comeron A., Kassianov E.I., Schafer K., Picard R.H., Weber K., and Singh U.N., Eds., SPIE, 2020. doi: 10.1117/12.2574234.
24. J. Zhang, "A Model of Diversified Fiscal and Taxation System Based on Computer Network Technology," in *Lect. Notes Electr. Eng.*, Hung J.C., Chang J., and Pei Y., Eds., Springer Science and Business Media Deutschland GmbH, 2023, pp. 254–261. doi: 10.1007/978-981-99-2092-1_32.
25. A. B. Singer, *Practical C++ design: From programming to architecture*. in Practical C++ Design: From Programming to Architecture. Apress Media LLC, 2021, p. 287. doi: 10.1007/978-1-4842-7407-1.
26. V. Sarcar and P. Shimanthoor, *Design Patterns in C#: A Hands-on Guide with Real-world Examples, Second Edition*. in Design Patterns in C: a Hands-on Guide with Real-world Examples, Second Edition. Apress Media LLC, 2020, p. 640. doi: 10.1007/978-1-4842-6062-3.
27. T. Bodhuin, E. Guardabascio, and M. Tortorella, "Migrating COBOL systems to the Web by using the MVC design pattern," in *Proc. Working Conf. Rev. Eng. WCRE*, Burd E. and van Deursen A., Eds., IEEE Computer Society, 2002, pp. 329–338. doi: 10.1109/WCRE.2002.1173090.
28. B. E. John, L. Bass, M.-I. Sanchez-Segura, and R. J. Adams, "Bringing usability concerns to the design of software architecture," in *Lect. Notes Comput. Sci.*, Springer Verlag, 2005, pp. 1–19. doi: 10.1007/11431879_1.
29. L. Cheng, W.-C. Niu, X.-G. Zhao, C.-L. Xu, and Z.-Y. Hou, "Design and implementation of college physics teaching platform based on virtual experiment scene," *Int J Electr Eng Educ*, 2021, doi: 10.1177/0020720920984688.
30. J. Andreeva *et al.*, "Designing and developing portable large-scale JavaScript web applications within the Experiment Dashboard framework," in *J. Phys. Conf. Ser.*, Institute of Physics Publishing, 2012. doi: 10.1088/1742-6596/396/5/052069.
31. P. Dewan, "Teaching inter-object design patterns to freshmen," in *Proc. Thirty Sixth SIGCSE Tech. Symp. Comput. Sci. Educ. SIGCSE*, Association for Computing Machinery, 2005, pp. 482–486. doi: 10.1145/1047124.1047498.
32. Y. H. Ding, C. H. Liu, and Y. X. Tang, "MVC pattern based on JAVA," in *Appl. Mech. Mater.*, 2012, pp. 537–541. doi: 10.4028/www.scientific.net/AMM.198-199.537.
33. W. Yong, "Design and Implementation of Intelligent English Electronic Dictionary System Based on Internet of Things," *Wireless Commun. Mobile Comput.*, vol. 2021, 2021, doi: 10.1155/2021/5586662.
34. X. Yu, X. Li, C. Wu, and G. Xu, "Design and Deployment of Django-based Housing Information Management System," in *J. Phys. Conf. Ser.*, Institute of Physics, 2022. doi: 10.1088/1742-6596/2425/1/012018.
35. X. Li, L. Wang, and L. Fang, "Optimal Design of an Information Management System for Government: A Bridge between Government and Citizens," *Math. Probl. Eng.*, vol. 2022, 2022, doi: 10.1155/2022/3127858.
36. P. Dewan, S. George, A. Wortas, and J. Do, "Techniques and tools for visually introducing freshmen to object-based thread abstractions," *J. Parallel Distrib. Comput.*, vol. 157, pp. 179–200, 2021, doi: 10.1016/j.jpdc.2021.05.013.
37. G. Li *et al.*, "Development of Data Visualization Interface in Smart Ship System," in *Lect. Notes Electr. Eng.*, Wang Y., Martinsen K., Yu T., and Wang K., Eds., Springer, 2020, pp. 219–226. doi: 10.1007/978-981-15-2341-0_27.

38. M. Singh and G. S. Walia, "Using Semantic Analysis and Graph Mining Approaches to Support Software Fault Fixation," in *Proc. - IEEE Int. Symp. Softw. Reliab. Eng. Workshops, ISSREW*, Vieira M., Madeira H., Antunes N., and Zheng Z., Eds., Institute of Electrical and Electronics Engineers Inc., 2020, pp. 43–48. doi: 10.1109/ISSREW51248.2020.00035.
39. P. Sauter, G. Vogler, G. Specht, and T. Flor, "Extending the MVC Design Pattern towards a Task-Oriented Development Approach for Pervasive Computing Applications," *Lect. Notes Comput. Sci.*, vol. 2981, pp. 309–321, 2004, doi: 10.1007/978-3-540-24714-2_23.
40. P. Sauter, G. Vögler, G. Specht, and T. Flor, "A Model-View-Controller extension for pervasive multi-client user interfaces," *Pers. Ubiquitous Comp.*, vol. 9, no. 2, pp. 100–107, 2005, doi: 10.1007/s00779-004-0314-7.
41. H. Jiang *et al.*, "DupHunter: Detecting Duplicate Pull Requests in Fork-Based Development," *IEEE Trans Software Eng*, vol. 49, no. 4, pp. 2920–2940, 2023, doi: 10.1109/TSE.2023.3235942.
42. L. Furtado, B. Miranda, N. Neto, and B. Meiguins, "IVOrpheus: A proposal for interaction by voice commands in three-dimensional environments of information visualization," in *Proc. - IEEE Int. Conf. Comput. Inf. Technol., CIT, IEEE Int. Conf. Ubiquitous Comput. Commun., IUCC, IEEE Int. Conf. Dependable, Auton. Secur. Comput., DASC, IEEE Int. Conf. Pervasive Intell. Comput., PICom*, Atzori L., Jin X., Jarvis S., Liu L., Calvo R.A., Hu J., Min G., Georgalas N., and Wu Y., Eds., Institute of Electrical and Electronics Engineers Inc., 2015, pp. 878–883. doi: 10.1109/CIT/IUCC/DASC/PICOM.2015.131.
43. B. Taraghi and M. Ebner, "A simple MVC framework for widget development," in *CEUR Workshop Proc.*, 2010. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-84889020688&partnerID=40&md5=e5c8427947e896cc262a6a4e0837c92e
44. S. Caballe, J. A. Ortega, J. M. Camps, L. Barolli, E. Kulla, and E. Spaho, "A presentation framework to simplify the development of Java EE application thin clients," in *Proc. - Int. Conf. Complex, Intelligent Softw. Intensive Syst., CISIS*, Institute of Electrical and Electronics Engineers Inc., 2014, pp. 421–426. doi: 10.1109/CISIS.2014.59.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.