International Journal for Innovative Engineering and Management Research
PEER REVIEWED OPEN ACCESS INTERNATIONAL JOURNAL
www.ijiemr.org

MODERNIZING ECOMMERCE: MONOLITHIC VS HEADLESS VS COMPOSABLE COMMERCE AND MICRO-FRONTENDS

Raghavendar Rao Tadpatri
Independent Researcher, USA.

Abstract - The growing pressure to modernize was piling pressure on many enterprises that were operating on IBM WebSphere Commerce (WCS) because of the rapid competition in digital commerce, the growth of multi-channels, or rising expectations around experience agility. Analyst commentary in and around digital commerce sites pointed to the momentum in the market, reported market growth and growing platform differentiation, pointing to the necessity of unpausing release friction and enhancing speed-to-market. The present paper presents a comparison of four strategies that are employed in the contemporary programs of modernization: (1) the traditional monolithic commerce, (2) the headless commerce, (3) the composable commerce and micro-frontend experience design. It briefly compares/contrasts both, outlines typical shortcomings observed when WCS is used in its monolithic form, and suggests a workaround of a gradual path to faster delivery, less upgrade risk, and channels in the future; headless-first, then compose selectively.

Keywords - WCS, IBM WebSphere Commerce, headless commerce, composable commerce, micro-frontends, modernization, digital commerce platforms.

I. INTRODUCTION

The environment of the digital commodities platforms is under mounting stress that requires it to keep evolving, with the pressures of faster releases, more multi-channeling, and supporting more agile and customer-oriented experiences. In these dynamics, IBM WebSphere Commerce (WCS) which was traditionally characterized by a monolithic architecture is facing challenges that are inherently linked to these dynamics [1]. Therefore, businesses are forced to upgrade their online systems to remain viable in the competitive market.

A. Problem Statement

The traditional monolithic approach of WCS is a barrier to the rapid adaptation of UI to changes, its scale and adoption of modern digital technologies [2]. This paper will critically examine other modernization directions of WCS such as headless and composable commerce and micro-frontend architectures.

This report aims to evaluate the challenges and opportunities of modernization of IBM WebSphere Commerce (WCS) systems through a comparison of monolithic, headless, composable, and micro-frontend integration and provide a feasible modernization policy.

B. Research Objectives

- ***To evaluate the limitations of conventional WCS monolithic deployment processes***
- ***To assess the advantages of headless commerce for improving experience agility and speed-to-market***
- ***To demonstrate the strengths and limitations of compostable commerce***
- ***To identify when micro-frontends are and how they affect the UI delivery process***

C. Novel Contribution

The research contributes to the current academic debate on e-commerce modernization since it provides a comparison of monolithic, headless and composable architecture, alongside the emerging trend of micro-frontends. It is a practical, step-by-step modernization plan that is IBM Web Sphere Commerce (WCS) specific and predictive of a headless-first orientation. The study not only sheds some light on the limitations of the conventional monolithic systems but also provides operational insights that enable a faster time-to-market provision, greater scalability, and better user experience by selectivity.

II. MARKET CONTEXT

A. Digital Commerce Platform Growth

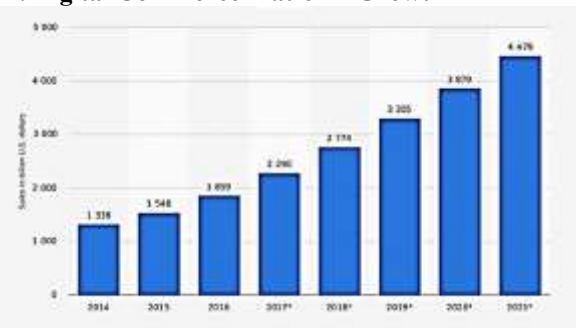

Fig. 1. Growth in Digital Commerce Platforms (2021)

The digital commerce platform market has experienced a growth of in 4.479 billion US dollars up to 2021, which highlights the need to upgrade the infrastructure by firms [3]. The Magic Quadrant of Digital Commerce Platforms by Gartner reveals that the market developed fast with a strong level of differentiation that occurs between the platforms, and

International Journal for Innovative Engineering and Management Research
PEER REVIEWED OPEN ACCESS INTERNATIONAL JOURNAL
www.ijiemr.org

thus, encourages enterprises to seek agile solutions that are API-driven and can facilitate quick release cycles and high scalability. The number below shows the rapid expansion of online trading platforms.

Fig. 2. Quadrant-style positioning concept (illustrative)

The above figure depicts how the digital commerce platforms are broken up into four quadrants: Visionaries, Leaders, Challengers, and Niche Players. Leaders are those who have good vision coupled with the ability to execute, whereas Visionaries are those who exercise innovation but have difficulty with execution. Challengers are masters of implementing but lack a big pictur,e and Niche Players are focused on a particular market, but are not big due to their lack of scalability and vision.

B. Shifting Architecture Paradigms

Fig. 3. The Forrester Wave™: Headless Content Management Systems (2022)

The development of the digital commerce platform is characterized by the abandonment of the idea of centralized architectures in favor of more versatile headless and buildable ones. These models contain the front and back-end decoupling, and this allows organizations to evolve rapidly at the front-end (UI) with consistent services at the back-end [4]. This shift is crucial in facilitating the experience of their experiences of different channels and enhancing the agility necessary to meet the changing consumer needs. The following chart indicates the increases in the use of headless commerce models.

C. Typical Limitations Observed with WCS Monolithic Deployments (2022)

Despite the number of digital commerce innovations increasing, many businesses still utilise WCS monolithic deployments. Such systems are subject to various impediments, which include: the inflexible connection between front end and back end changes, resistance to upgrade,s and loss of scalability [5].

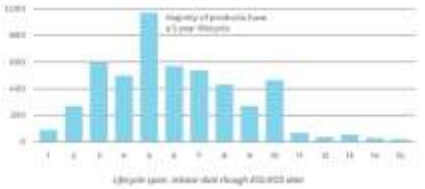

Fig. 4. Legacy System Modernization (2022)

Complete changes are a bad idea in a monolith system: when the UI gets changed, the back end will have to change as well, which will ratchet the release schedule. Moreover, the integration sprawl of monolithic systems such as point-to-point interfaces of disparate business applications, provides brittle dependencies that limit scalability and flexibility.

III. METHODOLOGY

This report emphasizes a comparative study using an analytical approach to evaluate four e-commerce architectures that include: Monolithic, Headless, Composable Commerce, and Micro-Frontends. The main purpose is to assess the weaknesses, advantages and suitability of each of the models to the modernization of IBM Web Sphere Commerce (WCS) deployments [6]. Theoretical modeling, system design, and performance evaluation using mathematical equations are used to quantify and compare the efficiency of both systems. As speed-to-market, scalability, communication agility, and complexity of integration, are the methodologies to be used.

Monolithic Commerce Model

Monolithic systems are characterized by a single deployable artifact where front-end (UI) and back-end (commerce services) are closely linked. In order to challenge the limitations of monolithic systems, we will focus on response time, upgrade unfriendliness, and scalability [7]. The time of response of monolithic systems is given as:

The response time for monolithic systems is expressed as:

$$T_{mono} = \frac{(N_{req} \times T_{processing})}{S_{resources}} \quad \text{----- (1)}$$

Her,e T_{mono} is the response time, N_{req} denotes the number of requests,$T_{processing}$ is the time taken to process each request, $S_{resources}$ denotes the available system resources. The performance of monolithic architectures will experience bottlenecks when there is an increase in the volume of requests, particularly

International Journal for Innovative Engineering and Management Research
PEER REVIEWED OPEN ACCESS INTERNATIONAL JOURNAL
www.ijiemr.org

because of the tightness between the UI and the back-end [8]. This presents severe restrictions on scalability and flexibility, especially to those enterprises that require frequent and swift updates.

Headless Commerce Model

Headless commerce decouples the front-end (UI) and the layer of commerce services with APIs, allowing both of them to be developed and deployed separately [9]. The design is agile to speed-to-market and experience. Headless commerce response time is modeled as follows:

$$T_{headless} = \frac{(N_{req} \times T_{processing}) + T_{API}}{S_{resources}} \quad \text{----- (2)}$$

Here, $T_{headless}$ denotes the response time for headless systems, T_{API} denoting the API response time for both UI and backend part. Headless commerce significantly reduces the release friction and allows organisations to provide personalized experiences on many channels at the same time [10]. It also provides the flexibility of constant experimentation of UI at the cost of the solid back-end commerce features.

Composable Commerce Model

Composable commerce builds on the concepts of the headless paradigm with requirements that can be independently replaced and then combined through APIs, modular components such as CMS, PIM, search, and personalisation [11]. This allows businesses to expand individual capacity without having to re-platform the whole system. Model of the performance of composable commerce is:

$$T_{composable} = \sum_{i=1}^{n} \left(\frac{N_{reqi} \times T_{processingi}}{S_{resourcesi}}\right) \quad \text{----- (3)}$$

Here, $T_{composable}$ is the total response time for these composable systems, N_{reqi} and $T_{processingi}$ refers to processing and requesting time for each modular component. $S_{resourcesi}$ representing the resource allocation for each component. Organisational flexibility can be enabled by component replacement or upgradation of compostable commerce. However, it also requires strong platform engineering and governance to address the details of integration, as well as avoid technical debt.

Micro-Frontend Model

Micro-frontends split the front-end into discrete, deployable units of user-experience that each have user-experience responsibilities, such as product details, search, and checkout. The trend encourages line-ups to form an independent deployment that enhances speed [12]. The modules are assigned to different teams, and this makes them efficient. There is a common design framework that ensures consistency of the UI. Independent deployments with CI/CD pipelines deliver through swifter deployments, the attachment includes a common nakedness handling that helps in routing and state [13].

IV. ARCHITECTURE APPROACHES: DEFINITIONS

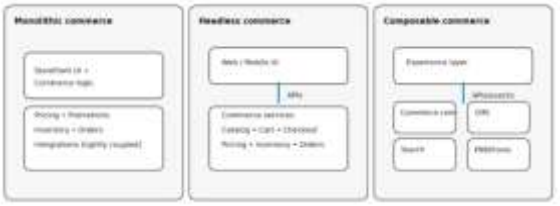

Fig. 5. Monolithic vs Headless vs Composable

In the context of modernization of e-commerce, there are a variety of architectural strategies offered to organizations. These architectures are explicitly designed to fulfil the requirements of improved scalability, faster time to market, and greater responsiveness in the user experience [14]. The following discussion outlines four major architectural paradigms, namely Monolithic Commerce, Headless Commerce, Composable Commerce, and Micro frontends.

A. Legacy Monolithic Commerce

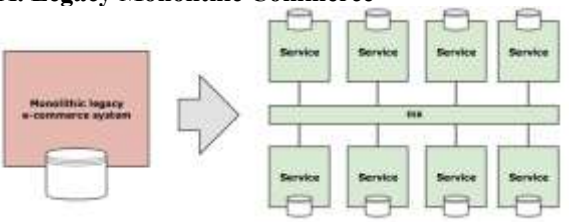

Fig. 6: Legacy Monolithic E-commerce

Traditionally, monolithic e-commerce systems such as one at IBM WebSphere Commerce (WCS) have been characterised by closely integrated front-end and back-end components. In spite of such architecture being simple and cost-effective to small businesses at the startup phase, the limitations become strong as the business grows [15]. Natural attachment between the back end and front end stretches release times as well as limits versatility hence harnessing responsiveness to the changing user needs and the electronic environment. Monolithic commerce is a significant inhibitor of organisational competitors in a setting in which competing rapidly is a mandatory factor.

B. Headless Commerce

International Journal for Innovative Engineering and Management Research
PEER REVIEWED OPEN ACCESS INTERNATIONAL JOURNAL
www.ijiemr.org

Fig. 7: Headless commerce architecture

Headless commerce separates the front-end user interface (UI) and the commerce services behind it, thus providing greater flexibility and faster time-to-market. The decoupling is enabled through application programming interfaces (APIs) that enable enterprises to test and customise the UI without significant damage to core back-end functionality of product catalogues, pricing and inventory management [16]. Therefore, headless commerce architectures are especially applicable to organisations with the need of multi-channel flexibility and rapid changes since the user-facing component can be updated separately on various platforms such as web-based web sites, mobile applications, and kiosks. Successful adoption of headless commerce by industry incumbents, including Nike and Amazon, shows empirical evidence of how these companies are able to quickly respond to customer demands and constantly adapt digital storefronts by the time of their release [17]. However, although headless commerce provides unprecedented nimbleness, it requires a powerful API management, substantial infrastructural underpinning, and tough API governance in order to create a seamless functioning inside a heterogeneous system.

C. Composable Commerce with Micro-Frontend Architecture

Fig. 8: Composable Commerce with Micro-Frontend Architecture

Composable commerce is based upon a modular architecture enabling the businesses to combine different best-of-breed solutions to different functions of commerce such as content management system (CMS), product information management tool (PIM), payment gateway, and loyalty programmes [18]. This modularity gives option to plug and play extensions meaning that the organisations customise and scale their systems without necessarily carrying out full platform migrations. Through APIs, businesses can constantly add or replace parts, e.g. by including subscription solutions or adding third party loyalty programmes.

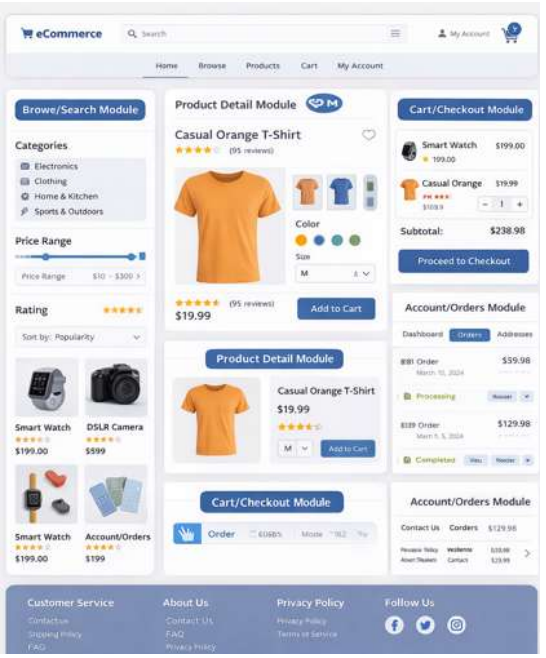

Fig. 9. Micro-frontends Design of E-commerce

Even though it improves flexibility, composable commerce requires strict platform engineering, along with careful governance, to reduce both the negative effects of integration sprawl and platform technical debt. When businesses become increasingly dynamic and reach greater scale, complex commerce makes it possible to integrate new solutions and maintain a strong and extensible scalable platform.

Micro-frontends are also additional to this one as they involve breaking down the front-end, then building smaller, deployable units that can be compiled and deployed independently by independent teams [19]. This structural separation facilitates simultaneous development, such that a group of thousands of people can develop (and grow) numerous features simultaneously in detail pages, search, and checkout. Nonetheless, although micro-frontends allow for creating accelerated development cycles, it is difficult to ensure coherence in the design of the UI of multiple components that are not in close contact with each other [20]. The challenge has been overcome by developing strict design systems and contract-testing procedures, thus having a way that all modules are made to conform to a single structure.

Use Cases for Micro-Frontend

International Journal for Innovative Engineering and Management Research
PEER REVIEWED OPEN ACCESS INTERNATIONAL JOURNAL
www.ijiemr.org

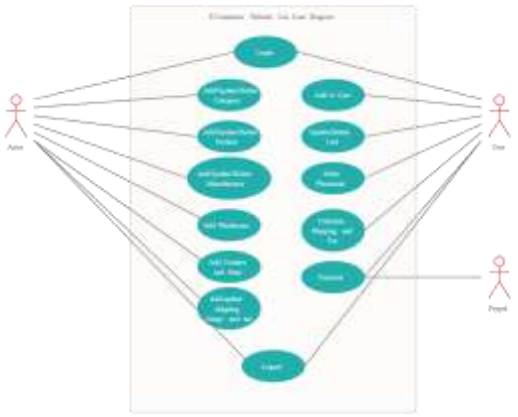

Fig. 10. E-commerce Use Case

The micro-frontends can also be of great benefit when a large monolithic storefront codebase fails to support multiple development teams. Micro-frontends can be used by enterprises that need to develop separately on different sets of features, including browse/search, product details, cart/checkout, and account/orders. Using the fragmentation of the frontend into smaller components that can be deployed independently, each of the teams can own and control a given feature area without having to wait until other teams are complete with their work.

Fig11.: E-commerce architecture evolution roadmap

TABLE 1: COMPARATIVE ANALYSIS FOR THE ABOVE THREE ARCHITECTURES

Architecture	Best Suited For	Advantages	Disadvantages	Ideal For
Monolithic Commerce	It is suitable for Small businesses, fewer channels and low change frequency process	Low initiation cost, simple setup.	Scalability issues and limited flexibility	Businesses with a low frequency of change
Headless Integration Model	Businesses that need agility, frequent UI changes, and multi-channel use	Front-end and back-end decoupling, faster time-to-market.	Requires strong API management, infrastructure, tand echnical expertise	Businesses with rapid digital transformation
Composable Commerce with Micro-Front-End	Large organizations, high customization, this model is used.	Modularity, flexibility, scalability, faster parallel development with micro-frontends	Complex integration, UI consistency challenges, and higher governance requirements	Large enterprises with high scalability needs

V. MICRO-FRONTEND DESIGN: WHEN IT HELPS AND WHEN IT HURTS

A. System Flow

Fig. 11. System Flow

The model of micro-frontend storefront decomposition, with a shared host application responsible for routing, authentication, design system, and analytics. It separates the frontend into autonomous modules, each being served by a different team: Browse/Search (Team A), Product Detail (Team B) Cart/Checkout (Team C) and Account/Orders (Team D).

C. Potential Drawbacks

International Journal for Innovative Engineering and Management Research
PEER REVIEWED OPEN ACCESS INTERNATIONAL JOURNAL
www.ijiemr.org

Although the implementation of micro-frontends may struggling create the complexity of ensuring consistency in terms of UX across modules. Having several teams working on various aspects of the UI it is often difficult to ensure the experience is seamless [21]. The design system's strict compliance, common UX standards, and contract testing are necessary to prevent fragmentation.

VI. EVALUATION OF APPROACHES

The analysis of the various eCommerce architectures, such as Monolithic architecture, Headless architecture, Composable Commerce, and Micro-Frontend, offers a good idea of their efficiency and appropriateness to upgrade the use of IBM Web sphere Commerce (WCS) implementations [22].

Although monolithic systems are less complicated to manage at first and less expensive with small businesses, they pose a constraint as the business expands. The close dependency between the front-end and back-end makes releases slower and makes it difficult to scale. This architecture struggles to integrate fast with frequent changes as well as the emerging technologies, which restrict the flexibility of multichannel integration.

Headless provides businesses with more flexibility and speed-to-market by making the separation of the front-end and back-end possible through APIs. It facilitates this by enabling quick prototyping of the user interface (UI) that will not impact central back-end processes [23]. Nonetheless, companies have to invest in good API management and a powerful back-end infrastructure.

Composable commerce brings a modular architecture, whereby the business can add limited-purpose components such CMS, PIM and search capabilities through APIs. This model has more flexibility, scalability, and customization. Some problems that arise in the form of lack of proper integration governance include integration sprawl among businesses.

The micro-frontend paradigm divides the UI into smaller modules that may be deployed independently, so that different teams may be developing different parts of the UI at the same time [24]. This approach allows accelerating the development rate and increasing autonomy but requires high-quality governance to assure a homogeneous user experience. Micro-frontends may pose a problem with the consistency of UI within teams and can add to the overhead of integration, so performance optimization is essential.

Finally, although both methods have their own advantages and disadvantages, headless commerce has shown the greatest advantage of speed-to-market and flexibility. Composable commerce is scalable over the long term, whereas micro-frontends are also an effective tool when team development requires concurrent development.

VII. PROS AND CONS SUMMARY

TABLE 2: PROS, CONS, AND BEST FIT

Architecture	Pros	Cons	Best Fit
Monolithic	Simpler early stages Fewer vendors Centralized operations	Slower change Upgrade friction Coarse scaling	Few channels Low change rate Centralized operations
Headless	UX agility Omnichannel APIs Isolated releases	More integration needed Observability needed API governance required	Multi-channel Fast UX iteration
Composable	Modular swap Best-of-breed Parallel teams	Higher complexity Vendor management issues Platform engineering	Large organizations Frequent change
Micro-frontends	UI parallelism Independent deployments Team ownership	Many UI teams Frontend bottleneck Shared UX discipline	UI parallelism Independent deployments

The table highlights the four eCommerce architectures that are Monolithic, Headless, Composable, and Micro-Frontends regarding their respective benefits,

shortcomings, and the most suitable to fit and be deployed in various types of businesses. A monolithic approach is simpler and centralized with a reduced rate of change as well as upgrade clash and reduced scales [25]. It is best applied in an environment that has a small number of channels and is characterized by low change rates.

Headless architecture enables more user experience (UX) agility and omnichannel through the separation of front and back-end services through APIs [26]. Nevertheless, it still needs more integration efforts and an easier way to observe so that it best suits organizations that focus on rapid experimentation of user experience across various channels.

Composable commerce is flexible in terms of the ability to replace modules and select the best-of-breed solutions, and it is thus ideal for large organizations and teams. Nonetheless, it may bring about complexity and challenges of vendor management and good platform engineering capabilities are required.

Micro-frontends have enormous benefits in the process of parallel development, with developers owning a particular front-end module [27]. As they offer independent deployments and ownership of teams, they pose a challenge in sustaining the consistency of theUI and need several UI teams to work together to prevent bottlenecks

VIII. RECOMMENDED PATH: A PRAGMATIC STRATEGY FOR WCS PROGRAMS

A. Start headless-first to unlock speed-to-market

In order to achieve faster time-to-market and enhanced user-experience responsiveness it would be recommended that organisations have a headless-first approach. The method separates the front and the back-end and allows more flexibility in the front-end changes, without interfering with the main commerce functionality. After the core systems are stable, businesses must then look forward to the micro-frontends, which allows the team to have different modules of the UI under their management independently and thus, faster development in parallel and deployment. Lastly, once operations stabilisation is achieved, organisations can selectively adopt composable commerce, using the best-of-breed elements where needed, or keeping strong platform engineering to handle complexity.

B. Use micro-frontends only when org structure demands it

The use of micro-frontends must be based on the organizational requirements, namely, in situations when the activities of large teams handle various aspects of the user interface (UI). The strategy will divide the front-end into smaller modules, which can be deployed separately, and each working team handles a different area of product details, search, or checkout [29]. Micro-frontends are therefore most appropriate in enterprises with a high level of distribution that can have distributed teams and the gains of running parallel development surpass the challenges of upholding consistency.

C. Compose selectively after stabilization

After a successful implementation of a headless architecture and the core systems are healthy, then the businesses should start to adopt composable commerce in a strategic manner. Businesses can use composable commerce to combine modular components, including CMS, PIM, and search systems, using APIs [30]. The nature of control and coordination of these elements, however, has to be complex, which means that it needs a foundation.

IX. CONCLUSION

In conclusion, organisations with traditional monolithic systems are faced with significant challenges in adapting to modern developments of the digital requirements. Headless commerce provides the needed flexibility and responsiveness to maintain pace with the quickly changing digital commerce environment and composable commerce provides organisations with the ability to enable their platforms to scale and tailor them accordingly. Micro-frontends may be further used to speed up the development and increase the autonomy of the team when properly applied through the strategic use of micro-frontends. A headless-first approach enables organisations to open up quicker release times; and micro-frontends and composable commerce provides long term scalability and agility. Future studies ought to look into the issue of integration between composable commerce and micro-frontends and explore their implications towards customer experience.

X. REFERENCE

[1] Vance, T.C., Wengren, M., Burger, E., Hernandez, D., Kearns, T., Medina-Lopez, E., Merati, N., O'brien, K., O'neil, J., Potemra, J.T. and Signell, R.P., 2019. From the oceans to the cloud: opportunities and challenges for data, models, computation and workflows. *Frontiers in Marine Science*, *6*, p.211.

[2] Giuliani, G., Masó, J., Mazzetti, P., Nativi, S. and Zabala, A., 2019. Paving the way to increased interoperability of earth observations data cubes. *Data*, *4*(3), p.113.

[3] Acs, Z.J., Song, A.K., Szerb, L., Audretsch, D.B. and Komlósi, É., 2021. The evolution of the global digital platform economy: 1971–2021. *Small Business Economics*, *57*(4), pp.1629-1659.

[4] Raja, J.Z., Chakkol, M., Johnson, M. and Beltagui, A., 2018. Organizing for servitization:

examining front-and back-end design configurations. *International Journal of Operations & Production Management*, *38*(1), pp.249-271.

[5] Vemuri, V.P.K., 2020. Addressing critical challenges in front-end performance and scalability. *International Journal of Multidisciplinary Research and Growth Evaluation*, *1*(5), pp.156-161.

[6] Jones, P., Wynn, M.G., Hillier, D. and Comfort, D., 2017. The sustainable development goals and information and communication technologies. *Indonesian Journal of Sustainability Accounting and Management*, *1*(1), pp.1-15.

[7] Kalske, M., Mäkitalo, N. and Mikkonen, T., 2017, June. Challenges when moving from monolith to microservice architecture. In *International Conference on Web Engineering* (pp. 32-47). Cham: Springer International Publishing.

[8] Mazzara, M., Dragoni, N., Bucchiarone, A., Giaretta, A., Larsen, S.T. and Dustdar, S., 2018. Microservices: Migration of a mission critical system. *IEEE Transactions on Services Computing*, *14*(5), pp.1464-1477.

[9] Shivakumar, S.K., 2020. Modern web platform performance principles. In *Modern Web Performance Optimization: Methods, Tools, and Patterns to Speed Up Digital Platforms* (pp. 105-143). Berkeley, CA: Apress.

[10] Roggeveen, A.L. and Sethuraman, R., 2020. Customer-interfacing retail technologies in 2020 & beyond: An integrative framework and research directions. *Journal of retailing*, *96*(3), pp.299-309.

[11] Mezzalira, L., 2021. *Building micro-frontends*. " O'Reilly Media, Inc.".

[12] Gilbert, J., 2021. *Software Architecture Patterns for Serverless Systems* (p. 436). Packt Publishing.

[13] Herron, D., 2020. *Node. js Web Development: Server-side web development made easy with Node 14 using practical examples*. Packt Publishing Ltd.

[14] Huang, D. and Wu, H., 2017. *Mobile cloud computing: foundations and service models*. Morgan Kaufmann.

[15] Giuliani, G., Masó, J., Mazzetti, P., Nativi, S. and Zabala, A., 2019. Paving the way to increased interoperability of earth observations data cubes. *Data*, *4*(3), p.113.

[16] Parsons, N. and Calabretta, N., 2020. Optical switching for data center networks. In *Springer Handbook of Optical Networks* (pp. 795-825). Cham: Springer International Publishing.

[17] Geers, M., 2020. *Micro frontends in action*. Simon and Schuster.

[18] Bastos, J.P.D.S., 2020. Desenvolvimento Web Orientado a Micro Frontends.

[19] Rappl, F. and Schottner, L., 2021. *The Art of Micro Frontends*. Packt Publishing.

[20] Vo, H., 2021. Applying microservice architecture with modern grpc api to scale up large and complex application.

[21] Rafael, S. and Ana, C., 2021. From Legacy Systems to Modern Frameworks: Leveraging Angular Elements for Cross-Framework Compatibility. *International Journal of Trend in Scientific Research and Development*, *5*(4), pp.1870-1880.

[22] Jumpponen, R., 2021. Modern Software Architecture.

[23] Shivakumar, S.K., 2020. Modern Web Performance Patterns. In *Modern Web Performance Optimization: Methods, Tools, and Patterns to Speed Up Digital Platforms* (pp. 273-300). Berkeley, CA: Apress.

[24] Prosper, J., 2019. Microservices Architecture for Agile Integration.

[25] Gilbert, J., 2021. *Software Architecture Patterns for Serverless Systems* (p. 436). Packt Publishing.

[26] Perez De Rosso, S., Jackson, D., Archie, M., Lao, C. and McNamara III, B.A., 2019, October. Declarative assembly of web applications from predefined concepts. In *Proceedings of the 2019 ACM SIGPLAN International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software* (pp. 79-93).

[27] Ribeiro, J.F.D.S., 2019. Análise e otimização de uma aplicação web.

[28] Kankaala, M., 2019. Enhancing E-commerce with Modern web technologies.

[29] Baidauletov, D. and Salehin, F., 2020. Integrating Modern Front-end Methodologies and Workflow in the Context of E-commerce Systems.

[30] Bernhard, M. and Mühling, T., 2020. E-commerce. In *Verantwortungsvolle KI im E-Commerce: Eine kurze Einführung in Verfahren der Künstlichen Intelligenz in der Webshop-Personalisierung* (pp. 67-118). Wiesbaden: Springer Fachmedien Wiesbaden.