Panamerican Mathematical Journal
ISSN: 1064-9735
Vol 34 No. 3 (2024)

Adaptive Middleware to Analyze and Confirm Data Centric Parameters for Data Aggregation in Federated Cloud Environment

Vikas K Kolekar[1], Sachin R Sakhare[2]
[1]Research Scholar, Department of Computer Engineering, SMT. Kashibai Navale College of Engineering, Pune.
[2]Professor, Department of Computer Engineering, Vishwakarma Institute of Information Technology, Pune.
Corresponding Author: Email: vikaskolekar2008@gmail.com

Article History:

Received: *02-06-2024*

Revised: *29-07-2024*

Accepted: *20-08-2024*

Abstract:

In the last decade rapid data evolution has contributed to tremendous data movement. The efficient data movement from origin to cloud storage and vice versa leads to the significant performance of application. A software layer situated between user and cloud storage called middleware plays major role in handling data to full fill user goals and requirements. In this paper, proposed a development of middleware frameworks from generic to context aware framework. Explicit importance is given to the adaptation feature of the middleware. A preliminary background is presented to list out importance and details of adaptability of the middleware and curious differentiation in static and self-adaptive adaptation features of the middleware. The literature started with the description of many generic middleware which can be used for building applications in variety of domains and further the limitations imposed are exposed. The issues like multidisciplinary data, vendor lock-in, disparity of service and interoperability related to federated cloud services serving to the fast-developing technology are highlighted. Along with that the use of context aware data management techniques for efficient handling of federated cloud system are defined. An extensive and comprehensive analysis of context-aware middleware is presented. For skilful data management processes need of context aware application with self-adaptation capability is presented especially in response to dynamic situations in federated cloud environment. At the end three techniques to design and develop context-aware middleware are presented. Alongside, four different perspectives of realizing adaptive middleware are also detailed. The presented work concludes with the highlighted need of context aware middleware framework to enhance the performance of federated cloud system.

Keywords: Adaptive Middleware, Data Centric Parameters, Data Management, Federated Cloud Environment.

1. Introduction

In this new era, emergence of all new technology is based on next generation connected devices. All these connected devices include numerous sensors and actuators, wearable electronic gadgets, radio frequency identification (RFID) tags and so many other devices. All this development rapidly fostering the evolution of enormous applications in variety of domains such as healthcare, home automation, transportation and logistics, traffic management, Industrial workflow management, manufacturing, smart-city, smart-grid, smart-agriculture and so on and so forth [1]. Each next generation connected device from all these domains acts as a data collection centre. As an overall scenario, numerous data is generated from all these things [2]. To manage this data and all the related applications cloud computing is emerged. At present many cloud service providers are available, due to which users have multiple choices for choosing the cloud service. However, in order to improve quality of service (QoS) and related cost optimization, the concept of federated cloud services is triggered. Whereas, more and more heterogeneous devices, technologies, and applications are getting connected in this ubiquitous computing environment and generating new challenges, such as interoperability, security, confidentiality, privacy, and energy-efficient operations for federated cloud system management [3]. For the efficient management of cloud computing, a software application, referred as the middleware is designed and developed. The middleware can be designed to hide the details of the things from applications while establishing communication between heterogeneous connected devices (things) by filtering and processing the raw captured data, before disseminating it further towards the connected applications, due to which, multiple common services [4] can be offered and development of backend applications is easier. Middleware can also be designed to tackle often arising issues such as heterogeneity, interoperability, security, and dependability [5]. Therefore, active ongoing research is required for middleware development which will cater today's necessities in terms of context awareness, scalability, device management, data storage and management, privacy, and service deployment. Designing of appropriate middleware solution in terms of the offered functionalities which satisfy the needs of applications and the underlying utilised technologies is a significant difficulty for application developers. Therefore, it is necessary to analyze the recent developments on middleware architecture in order to solve the technical concerns, difficulties, and gaps that are present in this field and recommend future improvements. The revolutionary development in the technology sector generated enormous amount of data and competitive availability of many cloud services throws up new challenges towards utilizing federated cloud storage systems. In the usual scenario cloud provider's services differs in the business model they offer which includes functional and non-functional aspects of the services, location and effective cost. This makes cloud provider selection process even more complex and the assortment of correct options for handling services in accordance with the application requirement becomes a challenging task. Therefore, in order to serve better QoS at optimal cost, appropriate designing of the middleware is utmost important.

However, manual management of the federated cloud storage is tedious since it involves assembly of heterogeneous resources from multiple clouds and moreover their properties are quick evolving. To handle this effectively, appropriate manual interventions in the form of continuous monitoring, consistent planning is required. Due to this complex situation, manual decision-making process for choosing a cloud provider and service that meets application requirements is complicated.

In the view of this, automation for such decision-making tasks can be a preferred choice through keen consideration of on-going changes in the cloud performance and effective costing. This is referred as adaptive middleware. However, adaptive decision making in case of federated cloud system can be more difficult since it has to be adaptive according to the QoS demands across the application lifecycle. Adaptive middleware can be designed by considering the functional or non-functional aspects and even multi-criteria. Designing of prediction model and planner are two major perspectives of middleware which significantly affects its performance. While featuring the adaptivity of the middleware, application's functional specifications (which are mainly resource details) and non-functional requirements (which take in to account QoS attributes) may be considered. Application's functional specification includes compute nodes (Virtual Machines (VMs)) along with their attributes such as power of CPU, size of RAM, the storage size and the data transfer rate. In usual cases these attributes are specified and less flexible, yet due to the availability of many resources, trade-off possibility still exists. However, in case of non-functional requirements lot many variations are flooding due to heterogeneity of user's requirement and available resources. In the view of this scenario, present review of the literature wherein research work is focused on QoS improvement of the middleware when non-functional attributes are considered in designing of prediction model and planner to build adaptive nature of the middleware.

The rest of the paper is organised as follows: Section 2 reviews the work related to middleware frameworks and context aware middleware. Section 3 discusses the identified research gaps and the overview of the developed middleware by providing brief insights about the middleware architecture and strategy used for its framework. Section 4 describes the context-aware middleware system to enable applications to adapt the changing contexts. Proposed the algorithm for precision-aware data management in federated cloud environments. Sample policies are proposed to implement the system. Section 5 presents the conclusion by highlighting the development of middleware frameworks from generic to context-aware framework. Section 6 provides the future directions for formation of context-aware middleware.

2. Background And Related Work

As per the requirements of the users, cloud computing serves in three primary segments. Software as a Service (SaaS) [6] makes different applications and infrastructure available (such as Application Program Interface (API)) for users at the initial stage. Platform as a Service (PaaS) makes platform available for developing complex applications such as Google Engine [6]. At last Infrastructure as a Service (IaaS) facilitates infrastructure services and capabilities for constructing and deploying VMs and storage facilities [7]. In order to cater variety of demands from the user's side, different approaches have been initiated for implementation of adaptive middleware [8]–[10]. Adaptive middleware signifies the process wherein behaviour of the middleware can be modified at runtime. In the meantime, emergence of numerous application domains, like cloud computing [11], IoT [12], [13], sensor networks [14], e-Health [15], and substantial development in newer processing techniques such as mining process [16], improved performance of model checkers, adaptation towards software architecture concepts [17], evolution of novel programming languages [18], are throwing new challenges towards development of adaptive middleware. However, designing of this kind of middleware firstly involves dealing with 5W1H issues accompanying with self-adaptive software [19].

2.1 Middleware Frameworks:

Ever evolving diversity of the users (related to many fields) and equipment increases the data handling complexity, due to which much more time and efforts are required for handling components like device-specification or domain-specification, and adaptive features like context awareness and uncertainty. For an instance, multiple intelligent systems must be able to determine situation based contextual changes from unintended behavioural changes of user and deliver suitable facilities while performing no explicitly separate operation, this lets us achieve extra with no significant efforts [20]. Along with the context-awareness property, situational context changes give rise to uncertainty, managing this is also one of the important significant challenges while featuring adaptive middleware. This is tricky since data aggregation from multiple systems on federated cloud platform cannot determine the diversity in the type and number of systems involved from consumers and cloud providers.

2.2 Context Aware Middleware:

A middleware, CAMPH [21] is proposed which connects hardware infrastructure to context-aware applications, notably ubiquitous homecare. This middleware provides context data gathering, storing, reasoning, query processing, service organization, and discovery. Whereas, four logical layers namely, physical space, context data management, service management, and application layers make up the architecture.

ACoMS+ [22] enhances ACoMS [23] to manage sensing infrastructure efficiently and contextually. It's core consists of a Context Source Manager (used for processing raw data and handling lower level communication), an Application Context Subscription Manager (is used for subscribing interested context by specifically mentioning information quality), and a Reconfiguration Manager (used to reconfigure sensing devices for availing the provision of fault-tolerance in information). For providing the ace of use of graphical notations for context information representation, Context Modelling Language (CML) is set as the modelling method. Use of Context Modelling Language (CML) as a modelling language leverages the graphical notations for contextual information representation ACoMS+ uses HiCoRE [24] mining to enhance reconfiguration. HiCoRE's mined context-related rules are used for wiser decisions. FIWARE [25] FP7 aims to boost EU competitiveness by providing a cutting-edge infrastructure for service creation and delivery, high QoS, and security. This general platform may be used in safety, logistics, environment, energy, transportation & mobility, and agriculture. This public cloud-based platform has a comprehensive module library with many value-added functionalities (referred as services). These Generic Enablers (GEs) provide service delivery, cloud hosting, Internet of Things, support services, development tools, and network and device interfaces for different chapters of this architecture. CA4IOT [26], is a middleware for sensing-as-a-service. Instead of offering a comprehensive middleware solution for handling context data, it seems that this middleware was just designed to address the specific problem of how to choose the more appropriate sensors in accordance with the activities or difficulties to be handled. This middleware carefully considers context abstraction, process, and dissemination; however, security and privacy are lacking. Unlike previous middleware, present middleware may be used alone or can be integrated with other frameworks to meet various paradigms' needs.

Compositional adaptation, ontology, and description logic/first-order logic reasoning is used in CAMPUS [27] to automate context-aware adaptation choices. Automated run-time adaptation judgments instead of predetermined adaptation procedures that only accept restricted contextual changes in a dynamic setting is a huge breakthrough made possible via this attempt. CAMPUS is a powerful middleware for developing context-aware application. Semantic-enhanced decision-making allowed CAMPUS to make runtime context-aware adaption choices. However, security and collaborative decision making across several CAMPUS middleware instances are referred as future scope of this.

Juyoung et al. proposed CASF [28], a context-aware middleware. Authors found that many context-aware middleware designs are lacking in service discovery and composition. To tackle this, semantic web services, which enable automated service discovery and composition capability, is used as a base model for further development. For selective combination of context services, this idea splits context-aware services and information and further integrates services with discovery and composition capability. The concept adoption from semantic web services enabled automatic discovery and integration of context information.

SeCoMan [29], which stands for Semantic Web-based Context Management, aims to provide a privacy-preserving approach for creating smart apps which are also context aware. SeCoMan uses ontologies to model entity description, analyze data to derive information, and construct context-aware rules. SeCoMan is a tri-layered structure which includes layers as Application, Context Management, and Plug-in. It contains selective location sharing feature due to which privacy protection is fulfilled. Forkan et al. [30] introduced CoCaMAAL, a Cloud-oriented Context-Aware Middleware for Ambient Assisted Living (AAL). AAL biological sensors lack processing capability to conduct crucial monitoring and data-aggregation activities. For this reason, cloud computing is employed to fulfil computational demands. This proposal is intended to provide a scalable, context-aware framework to AAL for data collection and processing. CoCaMAAL uses Service-Oriented Architecture (SOA) to manage and model raw context data and adaption, mapping context aware service, distributing service, and to discover service. Big Data for Context-aware Monitoring (BDCaM) [31] is a context-aware middleware architecture inspired by CoCaMAAL. BDCaM adds individualization in data analysis to CoCaMAAL and proposes multistep learning for context-aware decision making. It learns patient-specific abnormalities from data to obtain customised information. This new learning approach of BDCaM is significantly influencing its functionality towards giving personalized solutions or attention. On the other hand, due to this the security and privacy is compromised.

Few middleware(s) are cloud-based, implementable on many architectural styles, covers several different application domains, commercialized and utilising various infrastructures. A group of researchers proposed ABC (the Always Best Connected and best Served)) and S -based middleware for parking application [32]. This architecture consists of three layer as sensor layer, communication layer, and application layer. Amongst which application layer is responsible for taking cloud-based services.

Some self-adaptive cloud services are proposed previously. A group of researchers proposed three fuzzy algorithms for self-adaptive SaaS to avoid SLA (Service Level Agreement) breach in federated cloud system. The advantages of the present algorithms include dealing with three crucial issues: first QoS depends on low level metrics like uptime and downtime, second the service adaptation strategies are designed to handle business and IT infrastructure layers including changing time and complex computations, third, minimizing cost and influence of variations. Few researchers considered the necessity of ontology which converts generic concept to a higher level and depending on this proposed a context-based model CAMeOnto to provide mechanism for extending specific information using 5Ws principle (who, when, what, where and why). Some ontology-based context-aware modelling is proposed by few researchers which uses the concept of Semantic Web of Things (SWoT) [33], [34] and [35]. For the integration of heterogeneous smart objects for the monitoring and control of large-scale systems, a distributed framework is developed relating to Industry Middleware [36]. The context is modelled using key-value and ontology-based approaches, and the context is reasoned using rule-based techniques. The middleware offers features including virtualization of object, wireless sensor and actuator network (WSAN) gateways, distributed control logic, dynamic spectrum management, a SCADA gateway service, and data fusion transit capabilities.

Recently, Rafique et al. [37] proposed SCOPE, which is an autonomic middleware and used policy-based architecture. The SCOPE, is self-adaptiveness data management system for federated clouds and built using the standard Yahoo! Cloud Serving Benchmark (YCSB) [38] and the cassandra-stress tool. Even though it does not take in to account context awareness, the evaluation-based demonstration of the proposed middleware is presented highlighting the achievements as follows. Run-time dynamicity is accounted while managing data and corresponding SLAs are preserved. Independent of external interventions self-adaptivity is attainable. During this performance the acceptable performance overhead is also claimed. The various cloud trust model, middleware systems and algorithms has been proposed implemented to enhance user experience and cloud storage efficiency [39]-[44].

3. Research Gaps Identified

After extensive literature review as above of data management for federated cloud environment the following research gaps are identified:

- **Multidisciplinary Data:**

Heterogeneous data are generated from multiple sources and in multiple forms. Challenging to manage run time data getting generated from varied sources under single platform.

- **Vendor Lock-in:**

Data storage and accessibility feature varies from each service provider. It is now the user's responsibility to adapt the various interfaces and models whenever cloud user switch service providers to host their applications in the cloud. The adjustments that user expect in their service while switching to another provider would be complicated and chaotic due to the agreement with the specific service provider.

- **Interoperability Issue:**

Changing the cloud service provider with maintaining the service level agreement without affecting the any violations is difficult. Moreover, the norms and regulations of digital data handling differ across different countries. Perceiving all the directions in order to give and consume services in the global environment is a difficult.

- **Disparity of Services:**

Currently, the data being generated is often either interdisciplinary or multi-sensing in nature. Dynamic data that is generated is often saved on cloud platforms for convenient retrieval as needed. Cloud computing services are utilised to get proactive notifications. From the user's viewpoint, finding an appropriate cloud service provider for efficient storage and computing services may be highly tough. Users often lack knowledge about the best accessible solutions in terms of pricing, scalability, trustworthiness, and flexibility [44].

Middleware adaptation, in usual scenario is motivated by the continuous variations in application’s requirement, changes in surrounding conditions and bugs identified during functioning. Thus, eventually to serve users in better manner, middleware is needed to be improved for extending its functionality and performance. The enforcement of the adaptation must be determined by middleware designers, as well as when it is possible to do so. Additionally, it's important to decide whether adaptive efforts should be taken in response to undesirable behaviours or proactively to prevent them. For instance, the network should adapt if it gets overloaded, by forcing data compression (reactive).

Alternately, the adaptation process might occur when the mechanism of adaption notices that a middleware component's performance bottleneck can occur imminently (proactive). Identifying the areas where change is needed is a crucial component in designing an adaptive middleware. The middleware developer in this situation must decide if the adaptation must be made to a sole feature, a layer of middleware, or the entire architecture. The What problem, which is closely connected to the Where problem, calls for the definition of which artefacts or qualities must be altered when an adaptation is required. For instance, the modification might be as easy as replacing an algorithm or as complicated as adding or removing a middleware architectural component. Determining how the adaptation may be carried out and implemented is crucial. In general, adaption in the middleware is driven by the application's requirements and infrastructure conditions. Followed by this, adaption process must be critically designed by remembering the fact that during the adaptation process middleware itself can get affected and so the applications built atop it. For this, the current state of the application should also be taken in to account. In essence, formulation of adaptation framework or mechanism is critically important for the safer and performative acquisition of the adaptation. While, handling federated cloud management necessitates dealing with lack of standardization and imminent complexity during implementation. Adaptability of the middleware can be static or dynamic.

Table 1: Provides the overview of the developed middleware, provides brief insights about the middleware architecture and strategy used for its framework

Middleware	**Year**	**Context abstraction/Modelling**	**Context reasoning**	Context awareness	Interoperability	Service discovery	Adaptability
CAMPH [21]	2009	Key-value	Rule	Medium level	×	√	×
ACoMS+ [22]	2010	Graphical	Rule	Medium level	×	√	×
FIWARE [25]	2011	Ontology	Ontology	Semantic	√	√	×
CA4IoT [26]	2012	Ontology	Ontology and statistical	Semantic	√	√	×
CAMPUS [27]	2013	Ontology	Ontology	Semantic	√	√	Run time
CASF [28]	2013	Ontology	Ontology	Semantic	√	√	×
SeCoMan [29]	2014	Ontology	Ontology and rule	Location aware	√	×	×
CoCaMAAL [30]	2014	Ontology	Ontology	Semantic	√	√	Self-adaptation
BDCaM [31]	2015	Ontology	Ontology	Semantic	√	√	×
Industry middleware [36]	2015	Key-value, Ontology	Rule-based	Semantic	√	√	×
SAMSON [14]	2016	Ontology	Policy-based	Semantic	√	√	Run time

Cooperative middleware [42]	2017	Ontology	Rule-based, Ontology	Semantic	×	√	Dynamic
MidSHM [40]	2018	Not specified	Not specified	Not specified	√	√	Run time
SCOPE [37]	2019	Not specified	Policy-based (without Context consideration)	Not specified	×	√	Self-adaptation

Though, deployment of optimized dynamic methods is expected to produce better and higher adaptability, but it is hard to achieve in real-time. In contrast to flexibility, adaptability provides durability against consistent changes in requirement of the applications (from both user and owner of application). The process of changing structure or operation of an organization to increase its suitability in environment is referred as static adaptation. It makes adaptation of any service, application or system easier. For static adaptation some patterns are figured out previously to proactively handle context change. Because of this it is inefficient to handle dynamicity of object context. Whereas, self-adaptation addresses the issue of dynamic changes. Middleware framework in such cases need to be robust and flexible to increase the degree of autonomy. Autonomy refers to the adaptability of the system for changes occurred without any external interventions. Robustness is the system's ability to continue its services in the failure events and continue adaptation towards changes in the environment. To build an adaptation feature of middleware these things should be taken in to account, while the transparency should also get maintained for the user.

4. Proposed System

To bridge the above research gaps following system is proposed. In the progression of framing the adaptive feature in the middleware, one of the important aspects is to analyze and confirm federated data centric parameters for data aggregation in cloud environment. Listing out few data centric parameters includes performance, cost availability, resource discovery, response time, resource optimization and most importantly context awareness.

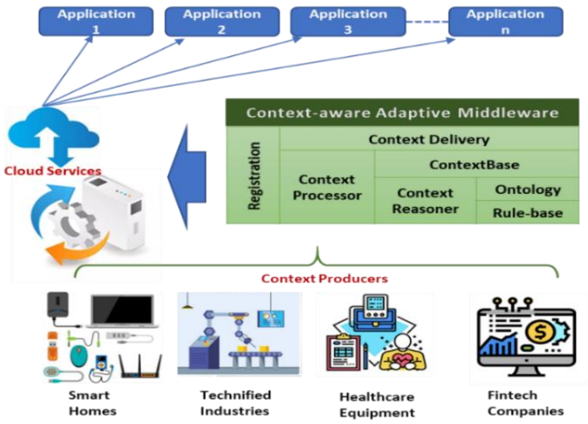

Figure 1: Proposed methodology for development of applications based on Context-aware adaptive middleware.

In essence, context-aware middleware is designed to enable applications to adapt to changing contexts, such as changes in network conditions, device capabilities, user preferences, and environmental factors, among others. Many different strategies can be employed to develop context-aware adaptive middleware and based on which numerous applications can be developed. Figure 1 elaborates overall usages of context-aware middleware. Several key components are involved in the working of context-aware middleware. It shows that large amount of data is collected from context producers such as smart homes, technified industries, healthcare equipment, and fintech companies. This data can include location, time, device type, network bandwidth, and other relevant information. The context delivery mechanism is responsible for transmitting the processed context data to the relevant components of the middleware, such as the context reasoner and rule-based system. It may use different communication protocols and formats depending on the context-aware application's requirements. It can also handle context updates and notifications. Context processor is responsible for collecting, interpreting, and transforming context data into a format that can be used by other components in the middleware. It gathers data from various sources, such as sensors, databases, and user input. The context reasoner is responsible for interpreting the context data and making decisions about how the middleware should behave based on the application's logic and rules. It may use a variety of techniques, including machine learning, rule-based systems, and fuzzy logic, to make informed decisions based on the context data.

Context ontology component defines a shared vocabulary and a set of concepts that are used to describe the context information. It provides a standardized way for different components to understand and interpret context data. Finally, the context rule-based system is responsible for enforcing the application's rules and policies based on the context data and the decisions made by the context reasoner. These rules can be defined by system administrators or end-users and can be used to trigger certain behaviours or actions. It may use a variety of techniques, including rule engines, expert systems, and constraint solvers, to ensure that the application behaves appropriately in different contexts. All together, these components form a powerful framework for building context-aware middleware that can adapt to changing environments and provide more personalized experiences for users. By incorporating context-awareness into software systems more intelligent, responsive, and efficient applications can be created that will efficiently meet the needs of users. Proposed the precision-aware data management algorithm for federated cloud environment considering the multidisciplinary data being generated from heterogenous sources. Federated cloud environments involve multiple cloud providers and data centers, and managing data in such environments can be challenging. In a precision-aware data management algorithm, the focus is on optimizing data placement and movement to improve performance and minimize costs, while taking into account the precision requirements of different applications.

The proposed algorithm for precision-aware data management in federated cloud environments will consist of following steps:

Step 1: Identify precision requirements.

Step 2: Determine data placement.

Step 3: Monitor data usage.

Step 4: Optimize data movement.

Step 5: Evaluate cost

The sample policies are proposed to implement the system. Policies that will act as intelligent based on data sets. So, adaptation of policies will be based on real time data sets.

Policy 1:

```
rule "Dealing in Public data"
when
    dsSelector : SmartHome (type= "PublicData")
then
    dsSelector.setConfidentiality(false);
dsSelector.readPreference("public");
dsSelector.writePreference("private");
dsSelector.storagePreference("public");
end
```

Policy 2:

```
rule "Dealing smart home authorized data with low Volume"
when
dsSelector:SmartHome(type= "SmartHomeAuthorizedData",  volume= "low")
then
    dsSelector.setConfidentiality(true);
    dsSelector.readPreference("private");
    dsSelector.writePreference("private");
    dsSelector.storagePreference("private");
end
```

To design context-aware middleware, three common techniques are proved quite helpful. First, contextual information is acquired, processed, interacted, and utilized by each application in a unique way. Second, some libraries and toolkits aimed at obtaining and processing contextual information may be supplemented and utilized again for developing context-aware application. Third, context aware application is built on the platform wherein context management is enabled. Since the third approach minimizes the efforts and complexity in development of context aware applications, it outperforms other two approaches. Taking this in to account, context aware middleware poses its strong candidature for building context aware applications.

However, adaptation in the middleware can take place in nearly four ways: First, without any external intervention, application autonomically adapts itself, second, overseen adaptation, wherein the adaptation process transparency is maintained by the platform for application, third, Structural adaptation, wherein program structure is changing, and fourth, Behavioural adaptation may be referred as functional adaptation which signify the behavioural changes in the program.

5. Future Directions For Formation Of Context-Aware Middleware

The context-aware middleware ideas are proliferated in recent years. However, in reality, degree of context awareness needs much more exploration for performance improvement. According to all available context information, the intended awareness implies that middleware might accurately grasp any change in the present environment. Most existing approaches for context-aware middleware only achieve a very basic degree of cognition and knowledge of their underlying environment. However, after careful consideration of the literature, following few influencing parameters, can be considered for improvement of the context-aware middleware performance. Traditional middleware, which serves as a software layer, is typically described as the metaphor of a black box, wherein hardware heterogeneity is masked and ease is provided for application development. Therefore, in order to differ, context-aware middleware should include basic context management, like modelling and processing of contextual information. Owing to this, developers can focus on the designing the more appropriate functionality and business logic of the application without concerning about the context management.

6. Conclusion

In this article, the development of middleware frameworks from generic to context-aware framework was discussed. Three popular methods for creating context-aware middleware were discussed. Alongside, four distinct approaches to implementing adaptive middleware are outlined. The middleware's adaptability feature is explicitly emphasized. First, a history is provided to outline the importance and details of the middleware's adaptability. Curiously, the middleware's characteristics for static and self-adaptive adaptation are also detailed.

The review of the literature began with a description of several generic middleware that may be used to create applications in a variety of domains, and it continued by highlighting the limitations imposed. The issues like multidisciplinary data, vendor lock-in, disparity of service and interoperability related federated cloud services serving to the fast-developing technology are highlighted and in line with that, use of context aware data management techniques for efficient handling of federated cloud system are described. It presents a thorough and comprehensive overview of the available context-aware middleware. Especially in response to dynamic conditions in federated cloud systems, skilful data management procedures explain the necessity for context-aware applications with self-adaptation capabilities.

References

[1] Mahmoud, E., Seyed, S., & Hon, C. (2017). Internet of Things Applications: Current and Future Development. https://doi.org/10.4018/978-1-5225-1832-7.ch015

[2] El Khaddar, M. A. (2021). Middleware Architecture (Middleware Solutions for the Internet of Things: A Survey). IntechOpen. https://doi.org/10.5772/INTECHOPEN.100348

[3] Issarny, V., Caporuscio, M., & Georgantas, N. (2007). A Perspective on the Future of Middleware-based Software Engineering. Future of Software Engineering (FOSE '07), Minneapolis, MN, USA, 244-258. https://doi.org/10.1109/FOSE.2007.2

[4] El Khaddar, M. A., Boulmalf, M., Harroud, H., & Elkoutbi, M. (2011). RFID Middleware Design and Architecture. InTech. https://doi.org/10.5772/16917

[5] Lee, S., Chang, J., & Lee, S.-G. (2011). Survey and trend analysis of context-aware systems. International Journal on

Information, 14(2).

[6] Cho, S., Hwang, S., Shin, W., Kim, N., & In, H. P. (2021). Design of Military Service Framework for Enabling Migration to Military SaaS Cloud Environment. Electronics, 2021(10), 572. https://doi.org/10.3390/electronics10050572

[7] Manvi, S. S. & Krishna Shyam, G. (2014). Resource management for Infrastructure as a Service (IaaS) in cloud computing: A survey. Journal of Network and Computer Applications, 41(1), 424-440. https://doi.org/10.1016/j.jnca.2013.10.004

[8] Blair, G., Coulson, G., Andersen, A., Blair, L., Clarke, M., Costa, F., Duran-Limon, H., Fitzpatrick, T., Johnston, L., Silva Moreira, R., Parlavantzas, N., & Saikoski, K. (2001). The Design and Implementation of Open ORB 2. IEEE Distributed Systems Online, 2.

[9] Clarke, M., Blair, G. S., Coulson, G., & Parlavantzas, N. (2001). An efficient component model for the construction of adaptive middleware. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 2218, 160-178. https://doi.org/10.1007/3-540-45518-3_9

[10] Kon, F. et al. (2000). Monitoring, Security, and Dynamic Configuration with the dynamicTAO Reflective ORB. In: Sventek, J., Coulson, G. (eds) Middleware 2000. Lecture Notes in Computer Science, vol. 1795. Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-45559-0_7

[11] Rafique, A., van Landuyt, D., Reniers, V., & Joosen, W. (2017). Towards an adaptive middleware for efficient multi-cloud data storage. CrossCloud 2017 - 4th Workshop on CrossCloud Infrastructures and Platforms, Colocated with EuroSys 2017. https://doi.org/10.1145/3069383.3069387

[12] Sarray, I., Ressouche, A., Gaffé, D., Tigli, J. Y., & Lavirotte, S. (2015). Safe composition in middleware for the internet of things. M4IOT 2015 - Proceedings of the 2nd ACM Workshop on Middleware for Context-Aware Applications in the IoT, in conjunction with ACM/IFIP/USENIX ACM International Middleware Conference, 7-12. https://doi.org/10.1145/2836127.2836131

[13] Park, S. & Song, J. (2016). Self-adaptive middleware framework for Internet of Things. The 4th IEEE Global Conference on Consumer Electronics, GCCE 2015, 81-82. https://doi.org/10.1109/GCCE.2015.7398741

[14] Portocarrero, J. M. T., Delicato, F. C., Pires, P. F., Rodrigues, T. C., & Batista, T. V. (2016). SAMSON: Self-adaptive middleware for wireless sensor networks. Proceedings of the ACM Symposium on Applied Computing, vol. 04-08-April-2016, 1315-1322. https://doi.org/10.1145/2851613.2851766

[15] Kliem, A., Boelke, A., Grohnert, A., & Traeder, N. (2014). Self-adaptive middleware for ubiquitous medical device integration. The 16th IEEE International Conference on e-Health Networking, Applications and Services, Healthcom 2014, 298-304. https://doi.org/10.1109/HealthCom.2014.7001858

[16] Rosa, N. (2017). Middleware adaptation through process mining. Proceedings - International Conference on Advanced Information Networking and Applications, AINA, 244-251. https://doi.org/10.1109/AINA.2017.25

[17] Rosa, N., Campos, G., & Cavalcanti, D. (2017). Using software architecture principles and lightweight formalisation to build adaptive middleware. Proceedings of the 16th Workshop on Adaptive and Reflective Middleware - ARM'17, 1-7. https://doi.org/10.1145/3152881.3152882

[18] Rosa, N. S., Campos, G. M. M., & Cavalcanti, D. J. M. (2019). Lightweight formalisation of adaptive middleware. Journal of Systems Architecture, 97, 54-64. https://doi.org/10.1016/j.sysarc.2018.12.002

[19] Sarray, I., Ressouche, A., Gaffé, D., Tigli, J. Y., & Lavirotte, S. (2015). Safe composition in middleware for the internet of things. M4IOT 2015 - Proceedings of the 2nd ACM Workshop on Middleware for Context-Aware Applications in the IoT, in conjunction with ACM/IFIP/USENIX ACM International Middleware Conference, 7-12. https://doi.org/10.1145/2836127.2836131

[20] Sousa, J. P. & Garlan, D. (2002). Aura: An Architectural Framework for User Mobility in Ubiquitous Computing Environments. In: Bosch, J., Gentleman, M., Hofmeister, C., Kuusela, J. (eds) Software Architecture. WICSA 2002. IFIP — The International Federation for Information Processing, vol 97. Springer, Boston, MA. https://doi.org/10.1007/978-0-387-35607-5_2

[21] Pung, H. et al. (2009). Context-aware middleware for pervasive elderly homecare. IEEE Journal on Selected Areas in Communications, 27(4), 510-524. https://doi.org/10.1109/JSAC.2009.090513

[22] Hu, P., Chong, S. K., Indulska, J., & Krishnaswamy, S. (2010). Context-aware and resource efficient sensing infrastructure for context-aware applications. The 8th IEEE International Conference on Pervasive Computing and Communications Workshops, PERCOM Workshops 2010, 492-498.

https://doi.org/10.1109/PERCOMW.2010.5470621

[23] Hu, P., Indulska, J., & Robinson, R. (2008). An autonomic context management system for pervasive computing. The 6th Annual IEEE International Conference on Pervasive Computing and Communications, PerCom 2008, 213-223. https://doi.org/10.1109/PERCOM.2008.56

[24] Chong, S. K., Gaber, M. M., Loke, S. W., & Krishnaswamy, S. (2008). ARTS: Adaptive Rule Triggers on Sensors for Energy Conservation in Applications using Coarse-Granularity Data. 2008 International Conference on Embedded Software and Systems, 314-321. https://doi.org/10.1109/ICESS.2008.57

[25] FIWARE | Open Source Platform for the Smart Digital Future. https://www.fiware.org/ (accessed November 13, 2023).

[26] Perera, C., Zaslavsky, A., Christen, P., & Georgakopoulos, D. (2012). CA4IOT: Context Awareness for Internet of Things. 2012 IEEE International Conference on Green Computing and Communications, Besancon, France, 775-782. https://doi.org/10.1109/GreenCom.2012.128

[27] Wei, E. J. Y. & Chan, A. T. S. (2013). CAMPUS: A middleware for automated context-aware adaptation decision making at run time. Pervasive Mob Comput, 9(1), 35-56. https://doi.org/10.1016/j.pmcj.2011.10.002

[28] Kang, J. & Park, S. (2013). Context-Aware Services Framework Based on Semantic Web Services for Automatic Discovery and Integration of Context. International Journal of Advancements in Computing Technology, 5(4). 439-448. https://doi.org/10.4156/ijact.vol5.issue4.52.

[29] Huertas Celdran, A., Garcia Clemente, F. J., Gil Perez, M., & Martinez Perez, G. (2016). SeCoMan: A Semantic-Aware Policy Framework for Developing Privacy-Preserving and Context-Aware Smart Applications. IEEE Syst J, 10(3), 1111-1124. https://doi.org/10.1109/JSYST.2013.2297707

[30] Forkan, A., Khalil, I., & Tari, Z. (2014). CoCaMAAL: A cloud-oriented context-aware middleware in ambient assisted living. Future Generation Computer Systems, 35, 114-127. https://doi.org/10.1016/j.future.2013.07.009

[31] Forkan, A. R. M., Khalil, I., Ibaida, A., & Tari, Z. (2015). BDCaM: Big Data for Context-Aware Monitoring—A Personalized Knowledge Discovery Framework for Assisted Healthcare. IEEE Transactions on Cloud Computing, 5(4), 628-641. https://doi.org/10.1109/TCC.2015.2440269

[32] Ji, Z., Ganchev, I., O'Droma, M., Zhao, L., & Zhang, X. (2014). A cloud-based car parking middleware for IoT-based smart cities: Design and implementation. Sensors (Switzerland), 14(12), 22372-22393. https://doi.org/10.3390/s141222372

[33] Jara, A. J., Olivieri, A. C., Bocchi, Y., Jung, M., Kastner, W., & Skarmeta, A. F. (2014). Semantic Web of things: An analysis of the application semantics for the IoT moving towards the IoT convergence. International Journal of Web and Grid Services, 10(2-3), 244-272. https://doi.org/10.1504/IJWGS.2014.060260

[34] Wang, X. H., Zhang, D. Q., Gu, T., & Pung, H. K. (2004). Ontology based context modeling and reasoning using OWL. Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops, Orlando, FL, USA, 18-22. https://doi.org/10.1109/PERCOMW.2004.1276898

[35] Ye J. et al. (2015). Semantic web technologies in pervasive computing: A survey and research roadmap. Pervasive Mob Comput, 23, 1-25. https://doi.org/10.1016/j.pmcj.2014.12.009

[36] Sadok, D. F. H., Gomes, L. L., Eisenhauer, M., & Kelner, J. (2015). A middleware for industry. Comput Ind, 71, 58-76. https://doi.org/10.1016/j.compind.2015.03.008

[37] Rafique, A., van Landuyt, D., Truyen, E., Reniers, V., & Joosen, W. (2019). SCOPE: self-adaptive and policy-based data management middleware for federated clouds. Journal of Internet Services and Applications, 10(1), 1-19. https://doi.org/10.1186/s13174-018-0101-8

[38] Cooper, B. F., Silberstein, A., Tam, E., Ramakrishnan, R., & Sears, R. (2010). Benchmarking cloud serving systems with YCSB. Proceedings of the 1st ACM Symposium on Cloud Computing, SoCC'10, 143-154. https://doi.org/10.1145/1807128.1807152

[39] Rathi, S. R. & Kolekar, V. K. (2018). Trust Model for Computing Security of Cloud. Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), Pune, India, 1-5. https://doi.org/10.1109/ICCUBEA.2018.8697881

[40] Y. Sahni, J. Cao and X. Liu, MidSHM: A Flexible Middleware for SHM Application Based on Service-Oriented Architecture, 2016 IEEE Symposium on Service-Oriented System Engineering (SOSE), Oxford, UK, 2016, pp. 126-135, doi: 10.1109/SOSE.2016.43.

[41] Kolekar, V. K. & Sakhare, S. R. (2021). Scheduling Algorithms of Cloud Computing: State of the Art. IJNGC, 12(2),

145-157.

[42] Hoque, M. Robiul & Kabir, M Humayun & Yang, Sung-Hyun. (2017). Development of a Cooperative Middleware to Provide Context-Aware Service in Smart Home. International Journal of Smart Home. 11. 33-40. 10.14257/ijsh.2017.11.5.03.

[43] Kolekar, V. K., Oswal, M., Wankhade, Y., Shirke, G., Sondur, A., & Wable, P. (2023). Smart City IoT Data Management with Proactive Middleware. IJRITCC, 11(5), 319-329. https://doi.org/10.17762/ijritcc.v11i5s.6754

[44] Kolekar, V. K. & Sakhare, S. R. (2023). A Research Perspective on Data Management Techniques for Federated Cloud Environment. IJRITCC, 11(5), 338-346. https://doi.org/10.17762/ijritcc.v11i5.6622