Computer Systems Science & Engineering
DOI:10.32604/csse.2022.021997
Article

Tech Science Press

Interactive Middleware Services for Heterogeneous Systems

Vasanthi Raghupathy[1,*], Osamah Ibrahim Khalaf[2], Carlos Andrés Tavera Romero[3], Sudhakar Sengan[4] and Dilip Kumar Sharma[5]

[1]Department of Computer Science and Engineering, Tagore Institute of Engineering and Technology, Attur, 636112, India
[2]Al-Nahrain University, Al-Nahrain Nanorenewable Energy Research Center, Baghdad, Iraq
[3]Universidad Santiago de Cali in Cali Colombia, South America
[4]Department of Computer Science and Engineering, PSN College of Engineering and Technology, Tirunelveli, 627152, India
[5]Department of Mathematics, Jaypee University of Engineering and Technology, Guna, 473226, India
*Corresponding Author: Vasanthi Raghupathy. Email: vasanti_me@yahoo.co.in
Received: 22 July 2021; Accepted: 23 August 2021

Abstract: Computing has become more invisible, widespread and ubiquitous since the inception of the Internet of Things (IoT) and Web of Things. Multiple devices that surround us meet user's requirements everywhere. Multiple Middleware Framework (MF) designs have come into existence because of the rapid development of interactive services in Heterogeneous Systems. This resulted in the delivery of interactive services throughout Heterogeneous Environments (HE). Users are given free navigation between devices in a widespread environment and continuously interact with each other from any chosen device. Numerous interactive devices with recent interactive platforms (for example, Smart Phones, Mobile Phones, Personal Computer (PC) and Personal Digital Assistant (PDA)) are available in the market. For easy access to information and services irrespective of the device used for working and even at the drastic change of the environment, the execution of applications on a broad spectrum of computing devices is propelled by the availability of the above-mentioned platforms. Different applications that need interoperability to coordinate and correspond with each other should be facilitated. Using a standard interface and data format, HE must link various devices from various platforms together to communicate with each other. To aid the interactive services performed by a middleware framework that operates on Application Programming Interface (API) over HEs, this issue aims to endorse an Adaptable Service Application Programming Interface (ASAPI).

Keywords: Middleware; services; ontology; interoperability

1 Introduction

There is increasing popularity for MFs, which perform interactive services over HEs because of the rapid development of interactive services in IoT and HE. Different platforms with distinct computing power, storage capacities, and user interfaces are already deployed in computing devices. Rather than decreasing, this kind of heterogeneity will increase over time since extensive use of new devices are expected.

Applications are developed at present for a certain kind of gadgets or system platforms, and the same applications are exclusively made for hand-held or desktops. They are otherwise personalized applications.

Moreover, for each device and processor family, applications should be typically disseminated and installed separately. Correspondingly, developing applications that are executed throughout all platforms will also be complex if the heterogeneity increases. Dissemination and application installation in every device and processor family are uncontrollable due to the increase in devices. So, a standard middleware, which can be used throughout HEs range of devices, is very much essential. The MF that works on API over HE is used by an ASAPI promoted by this problem for aiding interactive services.

2 Literature Review

The remote units, which do belong to the operating systems and hardware architectures, are given access by various middleware systems like Common Object Request Broker Architecture (CORBA), Electronic Data Interchange (EDI) and Distributed Component Object Model (DCOM). These middleware systems typically provide numerous functionalities resulting in complicated and resource-consuming systems not compatible with small portable and hand-held devices. For different features of mobile devices, semantic support can be adapted using an innovative framework and provides the mobile user's visibility on semantic functionalities enabled by nearby devices [1]. Distinct transformations of bytecode instructions are given with the help of supporting technologies like eXtensible Stylesheet Language (XSL) stylesheets and XML Path Language (XPath) expressions along with an eXtensible Markup Language (XML) representation [2]. The proposed and characterized semantic description of service requirements identifies process and service [3].

In distributed systems, interoperability is a fundamental and highly complicated issue about the discrepancy and dynamism displayed by modern systems [4]. The fulfilment of interoperability requirements of applications executed on multiple platforms becomes difficult due to various devices in heterogeneous computing environments. During the incorporation of financial systems and information exchange, the challenges for Semantic Interoperability (SI), a component-based approach for financial information, emerged [5]. The exchange of information and data on the web is facilitated by XML, a vendor-independent, media-independent, and platform-independent language [6]. XML is a simple and very flexible text format. The challenges of large-scale electronic publishing are met by the initially designed Standard Generalized Markup Language (SGML) from which XML is derived. Various kinds of data are elucidated by XML, and also a format was structured in a self-descriptive manner for storing and exchanging information. XML features an independent platform for depicting and data exchanging. Simultaneously, the format of XML data is both human and machine-readable. Like HyperText Markup Language (HTML), XML is not a fixed format but an extensible file format. XML is instead a meta-language for portraying other languages that can be designed for meeting the requirements of a specific domain such as science or business. Unicode is the primary base of XML files that gives a uniform representation irrespective of the file writer and reader's language. Generic XML schemas for business-to-business interoperability are provided by the XML data standards [7].

The acclimatization of the software entities to the changing situation is meant by Adaptability. Furthermore, adaptation becomes inevitable in case of a crucial mismatch between supply and demand [8]. In pervasive computing, adaptability is considered as one of the significant functionalities [9–13]. For instance, some application requires significant resources or services in the environment. The alternative resources in a new environment have to substitute the significant resources to guarantee continuous operation because the application's execution environment gets modified due to the user's mobility.

The new environment offers multiple network connections, and because of user's preferences, they qualify better. For instance, various devices like PC, PDAs and Mobile phones must be adaptable with

any services available in the Service Repository (SR). API specifies the medium through which these two systems communicate with each other. An application that permits the communication of other applications implements an API, which is a software-to-software interface. A set of functions, commands and protocols that achieves a particular task or communicate with a unique software element is specified by API. This set of functions or routines can be utilized by programmers while developing software for a particular operating system. Programmers are permitted to use predefined functions by API to communicate with the operating system rather than writing them from scratch. The applications developed by several vendors have no compatibility since there is no uniform development of software for devices. Without any user knowledge or intervention, applications talk to each other with APIs. By offering a homogeneity, API-like interface permits applications on various platforms or written in multi-languages for interoperation, middleware works.

According to the Institute of Electrical and Electronics Engineers (IEEE), interoperability is defined in a standard glossary of software engineering terminology as "the enabling of two or more systems or components for exchanging information and use one another's information." The cooperating subsystems need to interact and comprehend the information at the application level. The primary software provides the processing of a data representation at the system level. A vast range of programming prototypes, languages and enhancing environments is used by application developers and gives prediction regarding the persistence of heterogeneousness until expanding the range of uses for computing technology. In the future, different kinds of software components must support the infrastructure of HE, which is essential for the assimilation of software components. The software components may exist in distinct environments and compositions where effective interaction and cooperation can accomplish daily tasks.

The working together of diverse systems guarantees seamless exchange of information, an ongoing issue in distributed systems [14–16]. Some new problems, however, arise in HE and IoT development while achieving interoperability. Applications will be created gradually from the present software components because reacting to novel tasks and situations is essential for applications in HE. Dynamic interoperability is needed at the component level, which conquers the heterogeneity of the environment and components. In order to learn to communicate with the past anonymous components, they need to obtain each other's interfaces dynamically. The two kinds of middleware interoperability required by composition are discovering service and service communication protocols per dynamic application. The overpowering of semantic heterogeneity is related to the former, and it is addressed by the services and composite applications' semantic description. Fine mapping among heterogeneous middleware protocols is required by the latter [17–21].

The information exchanged by SI is meaningful and comprehensible. Besides semantics in data, self-described information packages are also added. SI guarantees the interoperability of IoT from various vendors. UI supplies data to IoT devices, and then, to make the data meaningful with shared vocabulary, semantic annotations from the semantic section are added. A language for explaining real-world semantics, which permits various changing interpretations and perfect automation to capture and create contextual metadata, is introduced in this paper. A mathematical semantics is already possessed by the language and aids SI's notion that it is a replica of the existing and formal notions of refinement. To understand the vision of IoT, interoperability, which is one of the critical challenges, needs to be addressed. Billions of devices will be allowed to interact and exchange information to complete tasks by attaining interoperability in IoT. Semantics are used to express the needs and capabilities of these devices and services offered by the devices in formalized ways. Semantic Web (SW) technologies explain the methods and tools that emerge from the field [22–26].

The enterprise information systems' interoperability mechanisms are implemented by a Semantic Web Services Oriented Architecture is proposed in paper. SWSOA is constructed on ontologies and contexts, and

for annotating web services descriptions, it denotes a reference model. Many fields like knowledge management, intellectual information incorporation, cooperative information system, and Database Integration use ontology that consists of notions, notion properties and the association between notions and restrictions. From the original data, ontologies are defined individually, and a general realization of the semantics of the discourse domain is replicated. A smart recursive acronym queries SPARQL Protocol and Resource Description Framework Query Language. The mining of resultantly merged data is visualized, and association rules on WEKA and Rapid minor software are generated. Ontology is developed using the Protégé tool. The models in different SW languages are edited by adapting knowledge acquisition [27–32].

Various features and intentions are focused on by having done many works on middleware. Using Message-Oriented Interactions, the existing Web Services Message Bus (WSMB), a less weight service-oriented framework, has improved interoperability. Through Messaging Services, each function in the functional interface based WSMB framework calls other functions. Lack of semantic transformation is the demerit of the existing WSMB framework, and to gratify the services in the heterogeneous computing environment, interoperability should be enhanced [33–37].

3 Conceptual Framework of Adaptable Service Application Programming Interface

In order that various PC, web and mobile applications communicate with distinct platforms, the proposed ASAPI model is designed. There is availability of vast range of interactive devices with new interactive platforms in the market. Applications are forced to work on a vast spectrum of computing devices, enabling users to access information and services irrespective of which device they work with and a dynamic change in the environment. There is a need for middleware with interoperability and semantic transformation to assimilate with various devices used for retrieving their functionality to assure a high level of user satisfaction. By sending requests to the middleware, applications specify their required functionalities. Contrarily, these requests can be explained with multi-languages or techniques. For the use of target system, the translation of request descriptions into an understandable system language is done [38–40].

Depending on the service request, the user interacts with the ASAPI model. According to the applications needed by the user, the ASAPI model reacts to the service requests by offering suitable services. Fig. 1 shows the conceptual framework of the ASAPI model. The configuration of several systems like PC, web and mobile applications is represented using various data organizations. SI is used besides the capacity of two or more computer systems to exchange information. To generate valuable results determined by the end-users of both systems, SI can automatically infer the meaningful and accurate exchange of information. The typical information exchange requires the reference of both sides, and to accomplish SI, XML transformation is used. By sending requests to the ASAPI model, the essential functionalities of the users are specified. Using multiple languages or techniques, these requests can be explained [41–43].

The four phases in the ASAPI model are as follows:

1. Organization Model
2. Application Manager
3. Matching Engine and
4. API Processor

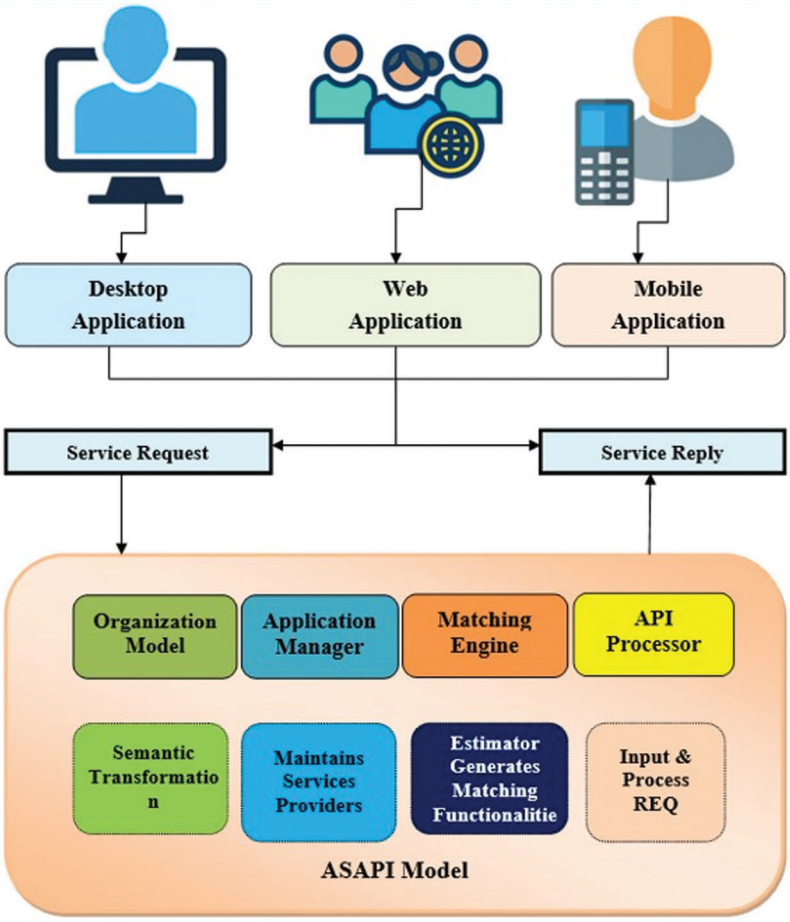

Figure 1: ASAPI model's conceptual framework

3.1 Adaptable Middleware Framework's Adaptable Service Application Programming Interface Model

The Organization Model (OM) of the ASAPI model is submitted with a service query requested by the users according to their needs. After parsing the user requesting queries into tokens in the OM, ontology is used to semantically transform these key terms that generally realise the term and its relation with other terms. Next, the semantic query is transmitted to the Matching Engine (ME). The Application Manager Phase and ME interact with each other. The AM phase maintains a list of available services in the SR. For each service available in the SR, the Metadata Repository (MR) maintains the Unique ID. The semantic query facilitates the interaction of ME with the AM Phase. First, the required service description in MR is searched by AM when it receives the Semantic Query. AM recovers its appropriate Services ID if it finds Search Result Collection to send it to ME.

The compare AM's Search Result Collection and the semantic query. AM transfers the obtained services to the Application Programming Interface Processor (APIP) if the obtained services are found accurately. Through an eXtensible Stylesheet Language Transformations (XSLT), the APIP semantic transformation of essential services with XML schema transforms it into API. Finally, the relevant API service(s) is provided based on the user's requirement, like desktop, web and mobile applications in the form of a plug-in, so that the APIP phase reacts to the user's request. Semantic query service is sent to Service Discovery Phase (SDP) if the former is mismatched. An SDP searches the User Requesting Query (URQ)

from the server and acquires the service sent to OM. The metadata information of the services is extracted in the OM and stores in MR. With a Unique Service_ID, similar services are stored in the SR.

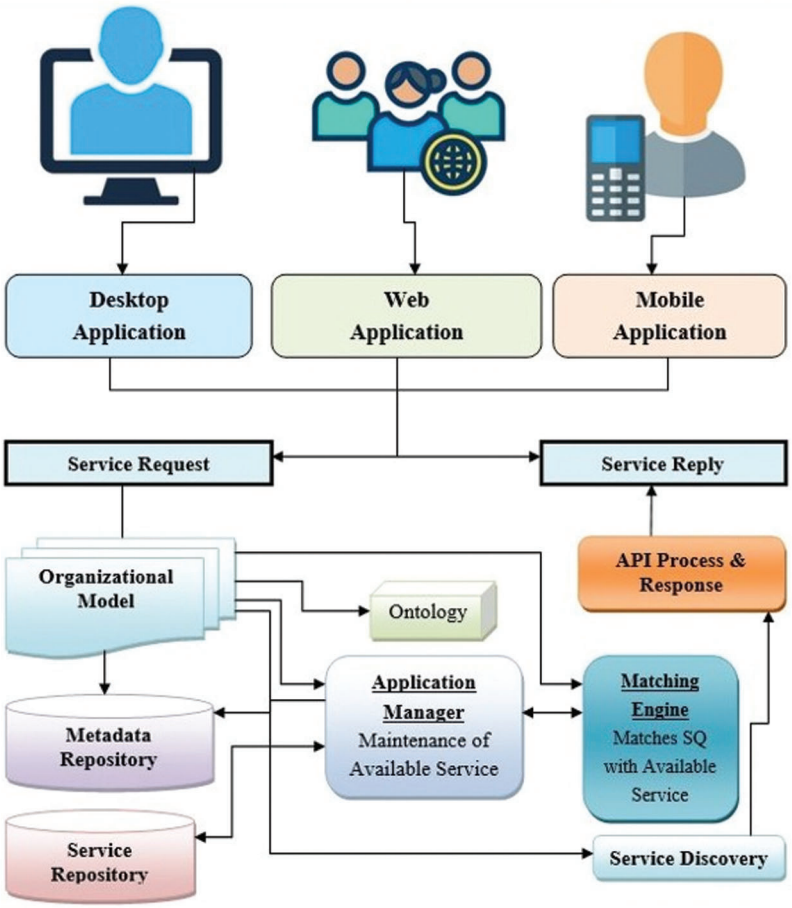

Figure 2: Shows the ASAPI model of adaptable middleware framework

The data to be stored in XML format is allowed by SR, which is an XML database. The desired format of these data is then queried, exported and serialized. The applications metadata information found in SR is stored in MR. For making domain-specific abstract model, an already available consistent set of terms and concepts is used. The already available set is represented and described explicitly by ontology. Information in properties that consist of values and similarities with the other properties is included in the abstract model.

The following are the steps to be followed in ASAPI Model Fig. 2

Step 1. Users provide URQ for the OM. The user is requesting the query's process semantic transformation into SQ.

Step 2. Adopt SQ from OM, and later, the AM Phase receives the SQ

Step 3. SQ is sent to AM by ME for searching in the available lists of MR

Step 4. A comparison is made between SW and the available lists in MR and the similar services of SQ obtained from SR is then send to ME.

Step 5. Compare the accurate match of SQ with the obtained services reverting from AM. The APIP otherwise goes to SDP if the available accurate match of services is transferred to APIP.

Step 6. Transfer the required services to the APIP.

Step 7. If the required service is unavailable, ME transfers its control to SDP.

Step 8. The required services are converted into an API response by APIP.

3.2 Adaptable Service Application Programming Interface Model's Algorithm

Fig. 3 Shows a flow chart of the ASAPI model. In ASAPI Model, the following steps are accomplished.

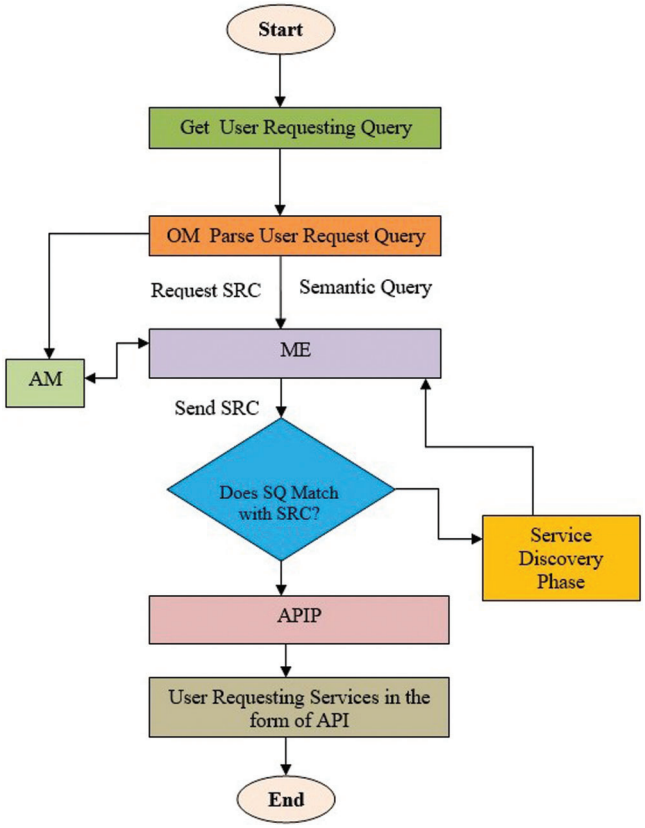

Figure 3: Organization of ASAPI model

Organization Model/ Semantic transformation of all applications*/*

Step 1. Obtain URQ in a HE

Step 2. Using an XML parser, parse the URQ into separate tokens.

Step 3. Using SPARQL query language, retrieve the semantic meaning of URQ from ontology.

Step 4. Send the SQ to ME.

Step 5. ME sends SQ to SDP if SQ is not an available service in AM.

Step 6. SQ is discovered from the server by SDP, and later, its application is sent to OM.

Step 7. Using XSLT, OM transforms the application semantically with XML Schema and sends it to AM.

Application Manager / Maintaining a List of Applications and Related services */*

Step 1. AM receives the semantically transformed application and its services from OM and stores in SR.

Step 2. Based on the semantic indexing method, the AM classifies the application.

Step 3. Send the SQ to the AM.

Step 4. The SQ received by AM is found in MR and stores Service-ID in SRC.

Step 5. SRC is returned to ME by AM.

Step 6. The process of matching is proceeded by ME.

Matching Engine / Generates Semantic Matching Functionalities */*

Fig. 4 represented the matching engine process

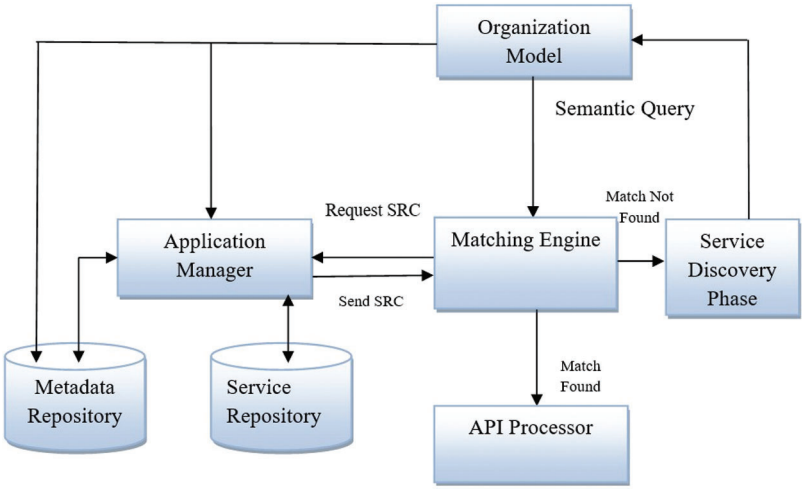

Figure 4: Matching engine process

Step 1. Obtain SQ from OM

Step 2. SQ is sent to AM by ME

Step 3. AM receives SQ

Step 4. For all keywords in SQ

Step 5. For all MR's service descriptions

Step 6. **IF** keyword and service description match with each other

Step 7. From an SR, retrieve the matched Service-ID and store it in SRC.

Step 8. SRC is returned to ME by AM

Step 9. **ELSE**

Step 10. SQ is sent to SDP by ME.

Step 11. The SQ application is found in the server by SDP, and the control is passed to OM

Step 12. The application is semantically transformed by OM and given to AM to categorize the application.

Step 13. **END IF**

API Processor / APIs Response to user */*

Step 1. User requested query's Matched Service (MS) is obtained from ME by APIP.

Step 2. The user request from OM is analyzed by APIP, in which the user query comes from the application platform, and the XML schema definition is taken according to that.

Step 3. The matched application services are converted by APIP, which are in the form of XML to API.

Step 4. API supports the concerned users to receive the user requesting application services in a Java plug-in.

4 Result and Discussion

Java has been implemented in the proposed ASAPI model. The Protégé ontology editor defines the ontology classes for the service description with the OWL language. XSLT uses XML Schema to perform semantic transformations. The performance evaluation test aims to compare the direct invocation by using HTTP *via* the ASAPI model.

The following metrics are performance evaluators of ASAPI:

a) Number of Application-User's requirement
b) Users Load Demand Variance-The memory usage of numerous services that is the user's need
c) Number of Required Services-The user's requirement of Number of Services
d) Number of Delivered Services-Delivery of Number of Services to the user
e) Binding Time-Time requirement for seeking a particular service

4.1 Binding Time Performance

While deploying the proposed model in a widespread computing environment, the interactive services' Binding Time has substantial improvement than the current WSMB. Fig. 5 shows the performance of Binding Time that makes an analogy between the ASAPI and WSMB model's performance in the aspect of metric Users Load Demand Variance measured in Kilobytes (KB) and Binding Time measured in milliseconds.

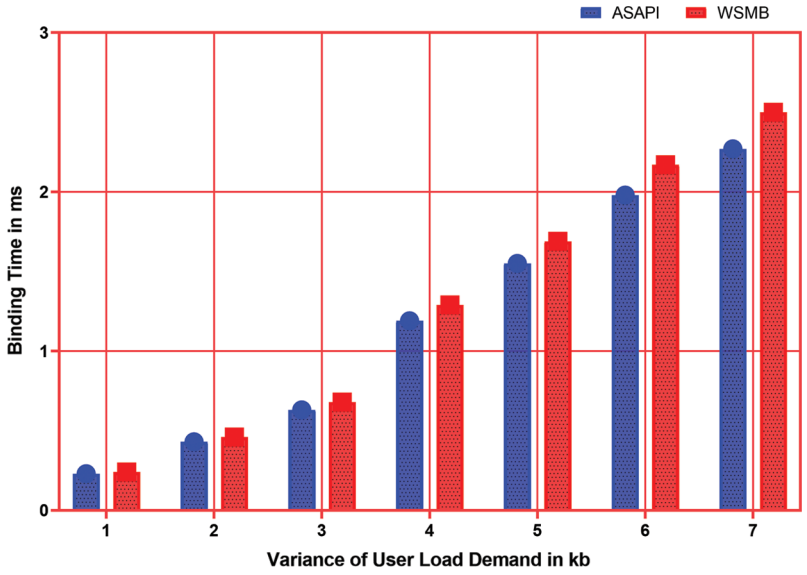

Figure 5: Binding time performance

Compared to the WSMB model, Binding Time's ASAPI model is reduced by 4–9% due to integrating function ontology with semantic transformation.

4.2 Performance of Delivered Number of Services

There has been an enhancement in the Number of Serviced Delivered performance over the WSMB model in a widespread environment in the proposed ASAPI model. This is because ME algorithm uses semantic ontology matching to minimize the Binding Time, successfully achieving synchronization and interoperability. The graph represented by Fig. 6 presents an analogy of the performance analysis of the proposed model and WSMB model in terms of the Number of Services Delivered. Compared to the WSMB model, Binding Time's ASAPI model is reduced by 8–11% due to integrating function ontology with semantic transformation.

Figure 6: Performance of delivered number of services

5 Conclusion

In this paper, the binding of multiple devices in a Heterogeneous Environment depending on the user demand is done by the essential services deployed as API. In this model, the interaction in a HE of the web, mobile and PC applications is facilitated using the flexibility of API interfaces with ontology and semantic transformation. The semantic transformation of applications by XML is used for interactive services of interoperable applications in a heterogeneous environment. The interactive service over HE is rendered by the proposed ASAPI model, and the enhancement of Binding Time is achieved using functional ontology in association with semantic transformation and Matching Engine Process. When compared to WSMB model, Binding Time is minimized by 4–9% in the proposed ASAPI model. By using semantic ontology matching, Binding Time with Matching Engine algorithm is minimized, and more than 8–11% interoperable services are delivered, thus achieving effective synchronization and enhancing interoperability.

Funding Statement: This research has been funded by Dirección General de Investigaciones of Universidad Santiago de Cali under call No. 01-2021.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] A. Saha, M. N. Tasdid and M. R. Rahman, "Mining semantic web-based ontological data," in *21st Int. Conf. of Computer and Information Technology (ICCIT)*, Dhaka, Bangladesh, pp. 1–5, 2018.

[2] B. Hu, K. He and J. Wang, "Semantic interoperability for financial information: A component-based approach," in *IEEE Int. Conf. on Computer Science and information Technology*, Chengdu, China, pp. 228–232, 2010.

[3] F. Lampathaki, S. Mouzakitis, G. Gionis, Y. Charalabidis and D. Askounis, "Business-to-business interoperability: A current review of XML data integration standards," *Computer Standards & Interfaces*, vol. 31, no. 6, pp. 1045–1055, 2009.

[4] G. S. Blair, A. Bennaceur, N. Georgantas, P. Grace, V. Issarny *et al.,* "The role of ontologies in emergent middleware: Supporting interoperability in complex distributed systems," *Middleware Springer Lecture Notes in Computer Science*, vol. 7049, pp. 410–430, 2011.

[5] J. Davies, J. Welch, D. Milward and S. Harris, "A formal, scalable approach to semantic interoperability," *Science of Computer Programming*, vol. 192, no. 1, pp. 102426, 2020.

[6] K. Mecheri, M. Boufaida, L. S. Meslati and W. Chabbi, "A framework for implementing the interoperability of semantic web services: Case study," in *Int. Conf. on Theoretical and Applicative Aspects of Computer Science (ICTAACS)*, Skikda, Algeria, pp. 1–8, 2019.

[7] K. Raatikainen, H. Christensen and T. Nakajima, "Application requirements for middleware for mobile and pervasive systems," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 6, no. 4, pp. 16–24, 2002.

[8] M. Satyanarayanan, "Pervasive computing: Vision and challenges," *IEEE Personal Communication*, vol. 8, no. 4, pp. 10–17, 2001.

[9] K. Mecheri, M. Boufaida, D. Meslati and L. S. Meslati, "Context-based interoperability of semantic web services," *International Journal of Metadata Semantics and Ontologies*, vol. 13, no. 3, pp. 209–222, 2019.

[10] P. Maheshwari, H. Tang and R. Liang, "Enhancing web services with message-oriented middleware," in *Proc. IEEE Int. Conf. on Web Services*, San Diego, CA, USA, pp. 524–531, 2004.

[11] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari and M. Ayyash, "Internet of things: A survey on enabling technologies protocols and applications," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.

[12] S. D. Nagowah, H. Ben Sta and B. A. Gobin Rahimbux, "An overview of semantic interoperability ontologies and frameworks for IoT," in *IEEE 6th Int. Conf. on Enterprise Systems*, Limassol, Cyprus, pp. 82–89, 2018.

[13] S. Han, Y. Bong and H. Youn, "A new middleware architecture for ubiquitous computing environment," in *Proc. of the 2nd IEEE Workshop on Software Technologies for Future Embedded and Ubiquitous Systems*, Austria, pp. 117–121, 2004.

[14] S. Jabbar, F. Ullah, S. Khalid, M. Khan and K. Han, "Semantic interoperability in heterogeneous IoT infrastructure for healthcare," *Wireless Communications and Mobile Computing*, vol. 2017, no. 1, pp. 1–10, 2017.

[15] V. Raychoudhurya, J. Cao, M. Kumar and D. Zhang, "Middleware for pervasive computing: A survey, pervasive and mobile computing," *Pervasive and Mobile Computing Journal*, vol. 8, no. 6, pp. 883–899, 2012.

[16] O. I. Khalaf, "Preface: Smart solutions in mathematical engineering and sciences theory," *Mathematics in Engineering, Science and Aerospace*, vol. 12, no. 1, pp. 1–4, 2021.

[17] O. I. Khalaf, M. Sokiyna, Y. Alotaibi, A. Alsufyani and S. Alghamdi, "Web attack detection using the input validation method: DPDA theory," *Computers, Materials & Continua*, vol. 68, no. 3, pp. 3167–3184, 2021.

[18] G. Suryanarayana, K. Chandran, O. I. Khalaf, Y. Alotaibi, A. Alsufyani *et al.,* "Accurate magnetic resonance image super-resolution using deep networks and Gaussian filtering in the stationary wavelet domain," *IEEE Access*, vol. 9, pp. 71406–71417, 2021.

[19] O. Wisesa, A. Andriansyah and O. I. Khalaf, "Prediction analysis for business to business (B2B) sales of telecommunication services using machine learning techniques," *Majlesi Journal of Electrical Engineering*, vol. 14, no. 4, pp. 145–153, 2020.

[20] A. T. Hoang, X. P. Nguyen, O. I. Khalaf, T. X. Tran, M. Q. Chau *et al.,* "Thermodynamic simulation on the change in phase for carburizing process," *Computers, Materials & Continua*, vol. 68, no. 1, pp. 1129–1145, 2021.

[21] X. Wang, J. Liu, O. I. Khalaf and Z. Liu, "Remote sensing monitoring method based on BDS-based maritime joint positioning model," *Computer Modeling in Engineering & Sciences*, vol. 127, no. 2, pp. 801–818, 2021.

[22] C. A. Tavera Romero, D. F. Castro, J. H. Ortiz, O. I. Khalaf and M. A. Vargas, "Synergy between circular economy and industry 4.0: A literature review," *Sustainability*, vol. 13, no. 8, pp. 4331, 2021.

[23] S. Sudhakar, G. R. K. Rao, O. I. Ibrahim Khalaf and M. Rajesh Babu, "Markov mathematical analysis for comprehensive real-time data-driven in healthcare," *Mathematics in Engineering, Science and Aerospace (MESA)*, vol. 12, no. 1, pp. 1–12, 2021.

[24] S. Sudhakar, P. Vidya Sagar, O. I. Khalaf and R. Dhanapal, "The optimization of reconfigured real-time datasets for improving classification performance of machine learning algorithms," *Mathematics in Engineering, Science and Aerospace (MESA)*, vol. 12, no. 1, pp. 1–14, 2021.

[25] G. M. Abdulsahib and O. I. Khalaf, "An improved cross-layer proactive congestion in wireless networks," *International Journal of Advances in Soft Computing and its Applications*, vol. 13, no. 1, pp. 178–192, 2021.

[26] X. Zheng, F. Ping, Y. Pu, C. E. M. Marin and O. I. Khalaf, "Recognize and regulate the importance of work-place emotions based on organizational adaptive emotion control," *Aggression and Violent Behavior*, vol. 23, no. 4, pp. 101557, 2021.

[27] M. Krichen, S. Mechti, R. Alroobaea, E. Said, P. Singh *et al.,* "A formal testing model for operating room control system using internet of things," *Computers Materials & Continua*, vol. 66, no. 3, pp. 2997–3011, 2021.

[28] O. I. Khalaf, K. A. Ogudo and M. Singh, "A fuzzy-based optimization technique for the energy and spectrum efficiencies trade-off in cognitive radio-enabled 5G network," *Symmetry*, vol. 3, no. 1, pp. 1–47, 2021.

[29] O. I. Khalaf, F. Ajesh, A. A. Hamad, G. N. Nguyen and D. N. Le, "Efficient dual-cooperative bait detection scheme for collaborative attackers on mobile ad-hoc networks," *IEEE Access*, vol. 8, pp. 227962–227969, 2020.

[30] A. A. Hamad, A. S. Al-Obeidi, E. H. Al-Taiy, O. I. Khalaf and D. Le, "Synchronization phenomena investigation of a new nonlinear dynamical system 4D by Gardano's and Lyapunov's methods," *Computers Materials & Continua*, vol. 66, no. 3, pp. 3311–3327, 2021.

[31] A. F. Subahi, Y. Alotaibi, O. I. Khalaf and F. Ajesh, "Packet drop battling mechanism for energy-aware detection in wireless networks," *Computers Materials and Continua*, vol. 66, no. 2, pp. 2077–2086, 2020.

[32] X. Xiang, Q. Li, S. Khan and O. I. Khalaf, "Urban water resource management for sustainable environment planning using artificial intelligence techniques," *Environmental Impact Assessment Review*, vol. 86, no. 19, pp. 106515, 2021.

[33] O. I. Khalaf and G. M. Abdulsahib, "Energy efficient routing and reliable data transmission protocol in WSN," *International Journal of Advances in Soft Computing and its Application*, vol. 12, no. 3, pp. 45–53, 2020.

[34] O. I. Khalaf, G. M. Abdulsahib and B. M. Sabbar, "Optimization of wireless sensor network coverage using the bee algorithm," *Journal of Information Science and Engineering*, vol. 36, no. 2, pp. 377–386, 2020.

[35] S. K. Prasad, J. Rachna, O. I. Khalaf and D. N. Le, "Map matching algorithm: Real-time location tracking for smart security application. telecommunications and radio engineering," *English translation of Elektrosvyaz and Radiotekhnika*, vol. 79, no. 13, pp. 1189–1203, 2020.

[36] O. I. Khalaf, G. M. Abdulsahib, H. D. Kasmaei and K. A. Ogudo, "A new algorithm on application of blockchain technology in live stream video transmissions and telecommunications," *International Journal of e-Collaboration (IJeC)*, vol. 16, no. 1, pp. 16–32, 2020.

[37] A. D. Salman, O. I. Khalaf and G. M. Abdulsahib, "An adaptive intelligent alarm system for wireless sensor network," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 15, no. 1, pp. 142–147, 2019.

[38] K. A. Ogudo, D. M. J. Muwawa, O. I. Khalaf and H. D. Kasmaei, "A device performance and data analytics concept for smartphones' IoT services and machine-type communication in cellular networks," *Symmetry*, vol. 11, no. 4, pp. 593–609, 2019.

[39] O. I. Khalaf and G. M. Abdulsahib, "Frequency estimation by the method of minimum mean squared error and P-value distributed in the wireless sensor network," *Journal of Information Science and Engineering*, vol. 35, no. 5, pp. 1099–1112, 2019.

[40] O. I. Khalaf and B. M. Sabbar, "An overview on wireless sensor networks and finding optimal location of nodes," *Periodicals of Engineering and Natural Sciences*, vol. 7, no. 3, pp. 1096–1101, 2019.

[41] G. M. Abdulsahib and O. I. Khalaf, "Comparison and evaluation of cloud processing models in cloud-based networks," *International Journal of Simulation—Systems, Science & Technology*, vol. 19, no. 5, pp. 26.1–26.6, 2018.

[42] O. I. Khalaf, G. M. Abdulsahib and M. Sadik, "A modified algorithm for improving lifetime WSN," *Journal of Engineering and Applied Sciences*, vol. 13, pp. 9277–9282, 2018.

[43] N. Sulaiman, G. Abdulsahib, O. I. Khalaf and M. N. Mohammed, "Effect of using different propagations of OLSR and DSDV routing protocols," in *Proc. of the IEEE Int. Conf. on Intelligent Systems Structuring and Simulation*, Langkawi, Malaysia, pp. 540–545, 2014.