

Analysis service architecture utilizing middleware for information services management systems

Suprihadi^{1*}, Sutarto Wijono², Kristoko Dwi Hartomo¹

¹ Department of Computer Science, Faculty of Information Technology, Satya Wacana Christian University, Salatiga, Indonesia

² Department of Psychology, Faculty of Psychology, Satya Wacana Christian University, Salatiga, Indonesia

ABSTRACT


The field of Information Service Management is concerned with the effective management of information technology (IT) services across their entire lifespan, encompassing activities such as design, development, deployment, operation, and continuous improvement. The components encompass many procedures, namely service design, service transition, service operation, and constant service improvement. The proliferation of information service management aligns with the ubiquity of the internet and its web-based services, notwithstanding the constraints posed by intermittent wireless connectivity. When the attainment of web service reliability is accomplished, there is a decrease in communication overhead, resulting in the retrieval of the correct response with little exertion. The worldwide pandemic issue presents an opportunity to glean useful insights that can contribute to the progress of digital health technologies. The remote monitoring of patient's health conditions and treatment actions inside the health system is crucial to mitigate the potential danger of disease transmission. Furthermore, there is a need for further development of digital technology that enables the sharing of information and facilitates evidence-based decision-making among stakeholders. The provision of data integration and analysis services is necessitated by the utilization of web services. The utilization of the middleware technique is deemed to be more suitable for web services. This study presents a novel model, referred to as Analysis Service Architecture utilizing Middleware (ASAM). The ASAM framework was established through the utilization of the Service Computing Systems Engineering Life Cycle approach. Further, ASAM is applied to the Management of Tuberculosis Information Services in Epidemic Tuberculosis cases in Indonesia. We evaluate and analyze our proposed method with other Information service management including face recognition, object recognition, and optical character recognition as a basis for evidence-based decision-making. Black box method testing with 50 service samples was carried out to measure aspects of functionality, reliability, and efficiency with ISO/ICE 9126 standards. The result is that the ASAM model is considered feasible as a new model of service-oriented middleware based on service-oriented architecture.

Keywords: Service architecture; ASAM; Information service; Artificial intelligence.

OPEN ACCESS

Received: July 17, 2023
Revised: October 13, 2023
Accepted: December 11, 2023

Corresponding Author:
Suprihadi
suprihadi@uksw.edu

 **Copyright:** The Author(s). This is an open access article distributed under the terms of the [Creative Commons Attribution License \(CC BY 4.0\)](https://creativecommons.org/licenses/by/4.0/), which permits unrestricted distribution provided the original author and source are cited.

Publisher:
[Chaoyang University of Technology](https://www.ceskaonline.com/)
ISSN: 1727-2394 (Print)
ISSN: 1727-7841 (Online)

1. INTRODUCTION

Information Services Management (ISM) refers to the practice of managing information technology (IT) and information systems within an organization to ensure their effective and efficient use in supporting business objectives and delivering value. It involves the planning, organizing, controlling, and coordinating of information resources, technologies, and services to meet the needs of an organization. Moreover, challenges to the application of digital technology are increasing. This is becoming increasingly clear considering the worldwide crisis, including the pandemic caused by Covid-19. The

Covid-19 pandemic has halted global efforts to reduce the burden of tuberculosis (TB) disease and so exacerbates the ongoing TB breakout crisis. The problem of the Covid-19 pandemic has caused a decline in the reporting of diagnosis and treatment of TB patients. Indonesia is one of the four contributing countries to global TB cases of 44%, experiencing a sizeable decrease of around 25–30% reporting the number of people diagnosed with TB between January and June 2020. This could lead to an increase in the number of TB deaths globally by around 0.2–0.4 million in 2020 to the number of people with global TB disease in the range of 8.9–11.0 million in 2019 (World Health Organization, 2020; Dewi et al., 2023a).

One of the information notes and guidelines of the World Health Organization (WHO) regarding TB and COVID-19 is to maximize the development of remote care and support for TB patients by using digital technology, to minimize the number of visits to health services, and realize community-based care (Phuttharak and Loke, 2023; World Health Organization, 2021). Research related to digital technology in efforts to eliminate the TB epidemic recommended by WHO (World Health Organization, 2017; Dewi et al., 2023c) includes short message service (SMS), electronic medication monitoring (EMM) (Wang et al., 2019), video-observed treatment (VOT) (Lee et al., 2020; Lyu et al., 2021). A research perspective for TB elimination where evidence of contribution achievements in the Sustainable Development Goals (SDGs) program is urgently needed by innovative digital technologies (Falzon et al., 2017). However, existing digital health technology has not had an optimal impact on TB patient care, so digital health technology research is still needed to improve patient treatment compliance by allocating resources wisely (Ngwatu et al., 2018). One of the studies related to digital health technology in the health sector related to TB is the development of a TB bacteria detection system based on digital image processing (Rohmah et al., 2019). From early 2016 to mid-2019, most of the digital technology research on TB focused on diagnostic tools and treatment adherence, much research is still needed to discuss data service technology and health systems or resource management (Lee et al., 2020). This has been proven by the end of 2022, remote-based healthcare systems have been developed by utilizing the internet of thing (IoT) and artificial intelligence (AI) (Teixeira et al., 2011; Bajaj et al., 2023), namely remote healthcare monitoring to increase performance efficiency, reduce costs, and improve the quality of patient care systems (Alshamrani, 2022; Dewi and Chen, 2022). In fact, efforts to improve the quality of patient care task scheduling have also been developed in the form of a priority-based task-scheduling and resource-allocation (PTS-RA) algorithm in a Health Monitoring System using mobile edge computing, as the basis for processing data from various IoT devices, which is a various sensor devices monitor the patient's health condition (Calimeri et al., 2019; Gamal et al., 2022; Sharif et al., 2023).

Furthermore, digital technology for TB elimination in

Indonesia developed by the Ministry of Health of the Republic of Indonesia is currently divided into 3 (three) services, namely specimen transportation tracking (SITRUST), TB patient assistance (EMPATI), and health information centers, list of health service facilities, sharing TB related community (SOBAT TB) (Findi, 2021). These three digital technology products have not been able to provide complete information for the completion of TB treatment and elimination because the problem of the TB epidemic is not only a health problem but also related to social and economic problems. The role of digital technology in TB care services and in the context of the Covid-19 pandemic, according to WHO, it is necessary to develop digital technology that can share information and make evidence-based decisions (Champaneria et al., 2023). Therefore, the need to share information and collect TB data from various data source technology modules scattered in various health facilities is urgently needed efforts to eliminate TB in the National Tuberculosis Program (NTP), including treatment adherence, epidemiological surveillance, and operational responses modules (Ardhianto et al., 2022). It is also necessary to develop Disruptive Tuberculosis IT to be able to build data integration and information management services needed by relevant stakeholders and the community besides the health facility (Riono, 2019; Dewi et al., 2023b).

1.1 State of the Art of Reliable Service Architecture Using Middleware (RSAM)

Integration technology from various data sources and applications is needed in the process of generating information. A new cloud computing-based service management paradigm and model has been developed to support modern enterprise business processes. Service applications can be built in the form of web services that provide functions and operations that can be accessed via network protocols and do not depend on specific platform technologies (Wu et al., 2015; Giau et al., 2022). Thus, web services can facilitate collaboration of business services between organizations that are exposed openly through the internet network. Web services as a basis for service computing technology have also proven their reliability in building a Reliable Service Architecture using Middleware (RSAM) (Abdelfattah et al., 2020). RSAM is intended to ensure and track request execution under communication limitations and temporary unavailability of services, so that the reliability of web services in mobile cloud computing can be achieved (Abdelfattah et al., 2018; Dewi et al., 2023d).

1.2 State of the Art of Service Oriented Using Middleware

One of the data integration technologies is Message Oriented Middleware (MOM). MOM is a service technology for exchanging metadata-based data sourced from the Data Base Management System (DBMS) and

Open Government Data (ODG) (Sukarsa et al., 2014; Sembiring and Uluwiyah, 2015; Wisswani and Wijaya, 2018). This Message Oriented Middleware (MOM) model is very reliable for data integration services in integration systems if the specified metadata meets the needs and can be provided by each connected user. As such, the MOM architecture is typically reserved for message exchange and data integration providers, so there is no guarantee of a service to perform data analysis.

Another integration technology is Service Oriented Middleware (SOM) (Lamnaour et al., 2022). Besides, SOM architecture is used to achieve interoperability and heterogeneity in data and system integration. In wireless sensor networks (WSN), the SOM architectural model is able to provide services to meet the needs of different domains, applications and network systems (Naseer et al., 2016). SOM was also developed to support SmartCityWare applications in the integration and utilization of Cloud of Things (CoT) and Fog Computing, so that they can develop and operate various smart city applications effectively (Mohamed et al., 2017). An SOM application is principally the implementation of a service computing system that involves interaction, composition, and collaboration between IT services, as well as interaction and dependency between users and service providers (Huang et al., 2013; Huang et al., 2014). Service computing system is also a group of IT service interaction as service computing and IT infrastructure which also pays attention to service performance, such as QoS and service discovery (Chen et al., 2017). In addition, service computing systems are intended to meet complex user requirements, namely composite services as an integration of individual services consumed by users (Ye et al. 2019; Kuhn Cuellar et al. 2022). SOM applications as a computing system service have been developed by many IT companies providing cloud computing services, such as Google, Microsoft Azure, and Amazon Web Service (AWS). But the service will be paid for if the usage exceeds a certain limit. This is an obstacle for government agencies in Indonesia, namely having a policy where the use of cloud computing services is free. In fact, analysis services are urgently needed in the form of Artificial Intelligent (AI) and machine learning modules that apply to system integration between stakeholders implementing the TB elimination program (Nayak et al., 2023).

1.3 State of the Art of Analysis Service Using Middleware

With the MOM and SOM approaches, the need for data analysis in the process of generating information and evidence-based decision making can be developed. Data analysis services can be developed from the SOM architectural model, while data integration needs to meet the availability of data in the data analysis module can be developed using the MOM approach. Analysis modules in the form of machine learning and deep learning can be

developed independently or utilize analysis modules provided by RapidMiner, Keras and Tensorflow which are open-source applications in the form of a javascript application programmable interface (API)(Dewi et al., 2021; Cao, 2022).

The following is a list of the most significant contributions that this article has made: (1) We propose an architectural design Analysis Service Architecture utilizing Middleware (ASAM) to manage dynamic information about TB case in Indonesia. The proposed ASAM approach uses a middleware approach but addresses a new and different case. (2) Our work integrated two important components, the analysis service module component, and the analysis result data exchange service module in the cloud service layer. This approach considers the need for patient-focused TB information management for various IT infrastructure resources owned by TB elimination program stakeholders. ASAM also proposes a new service architecture model that can collaborate several analysis modules and AI modules. With ASAM, it can help TB elimination programs in the treatment of TB patients by providing evidence-based information. (3) We evaluated and analyzed our proposed method with different service systems including the get by NIK other server service, Face Recognition Service, Object Recognition Service, and Face Recognition Service.

The following is the structure of this research work. The Materials and Methods section covers related work as well as the methodology we plan to apply in this research. Section Result and discussions describes our experiment setting and results. Finally, in the final section, conclusions are drawn and suggestions for future research are made.

2. MATERIALS AND METHODS

To understand the benefits of using ASAM, this study uses a system development method, namely the Service Computing Systems Engineering Life Cycle as shown in Fig. 1, because ASAM is principally a Service Computing Systems. The Service Computing Systems Engineering Life Cycle was chosen because the stages of this method are able to guarantee that ASAM is built according to the business service needs of an organization (Suhardi et al., 2017), in this paper, namely health facilities both government and private.

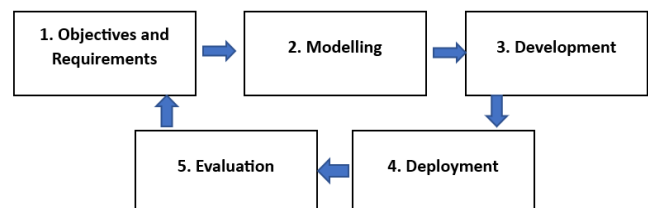


Fig. 1. Service computing systems engineering life cycle

The Community Health Center (Balkesmas) serves as the

data source for the aims and requirements stages. Located in Magelang, Central Java Province, this community hospital is run by the government. The health facility belongs to the Central Java provincial government under the management of the Ministry of Health of the Republic of Indonesia. Based on the results of interviews and observations related to the condition of the information system infrastructure owned by Balkemas, it is known that TB patient care data is still local in nature, there is no national TB patient care data center yet available. This is one of the causes of TB patient treatment failure, due to weak data and information services if the patient must move to a health facility. The modeling stage is the stage of compiling the data integration model and patient service system to meet needs and provide solutions to problems found during the goals and requirements stage. At the development stage, a web service technology was selected for data exchange services, and the architecture was developed to become a middleware capable of providing process data analysis services. The deployment stage is implementing the ASAM model on 3 (three) servers, and 1 (one) database server as a patient data center. This was chosen to be close to the existing IT infrastructure conditions in health facilities in the Central Java region. The evaluation stage is the stage of testing the feasibility of the ASAM model performance. These stages are carried out repeatedly until they meet the eligibility targets of ASAM. Therefore, the proposed ASAM can be a simple, inexpensive, and effective IT infrastructure to meet data integration needs, as well as provide information on the results of data analysis (Suprihadi et al., 2020).

2.1 Analysis Services Using Middleware Approach

SOM is a web application built to act as a data communications gateway between clients and servers. In principle, middleware is a web service. Web services built using the middleware approach have reliability, including having a higher response size and consumer time of only around 5% with direct cloud services for the same request size. In addition, web services with a middleware approach are also able to guarantee service communication between clients and cloud services (Abdelfattah et al., 2018; Castelli et al., 2021).

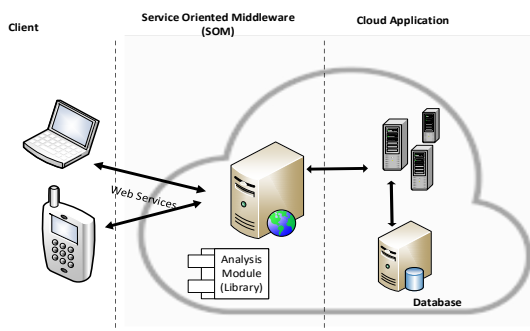


Fig. 2. SOM architecture

The SOM architecture, as shown in Fig. 2, consists of the following three layers:

Client: The architecture layer which can be a desktop, laptop, or mobile computer device that has internet access to use SOM (Prisilla et al., 2023).

SOM: Middleware is a web application that contains web service components consisting of service builder, analyst builder, query builder, dictionary, and library. Middleware with a SOM structure has been able to provide web services that provide processing services for data analysis, or query data from databases. For this reason, SOM also provides a library containing analysis modules. Web services on SOM are built using the RESTful method approach using the CodeIgniter (CI) framework.

Cloud Application: The layer that contains applications that function to manage databases, and other services used in the SOM layer. In this study using a database engine, namely MongoDB (Tjung et al., 2020).

The next step is the Modeling stage, namely designing a system requirements analysis model to be built. The analysis model uses activity diagrams to understand the service process flow using ASAM.

2.2 Proposed Analysis Service Architecture utilizing Middleware (ASAM)

A new ASAM is offered, focused on improving web service functions by adding data analysis services that are integrated with cloud applications as middleware. ASAM has two main components, namely SOM component (SOMC) and cloud application component (CAC) as cloud services that reside in the cloud layer providing services to client applications. Client applications as ASAM consumers, are applications that are permitted to establish communication with the CAC as cloud services directly or through SOMC. The client application is required to build a request with the appropriate attributes and format to be sent to obtain or receive a response. The ASAM model activity diagram can be seen in Fig. 3 which shows the process flow from the request service to the response service. Patient authentication services are carried out in 2 (two) stages, namely based on NIK and facial recognition. Face recognition was chosen to reduce the risk of transmission by touch. If NIK data is not found in the database of health facilities visited by patients, ASAM will look for NIK data in other registered health facilities. This activity is a proposed service mechanism to help improve TB patient care services in Indonesia. Further, ASAM and its components are implemented in every health facility. Thus, TB patients can obtain appropriate care services at each health facility. Evidence-based drug monitoring is proposed for use with other services, like object recognition and OCR, in patient care. This was recommended as an alternative to checking in with patients upon their readmission to determine if they had taken their prescribed medications.

SOMC (Mesmoudi et al., 2020; Zhang et al., 2022), is a middleware in the form of a RESTful web service provided

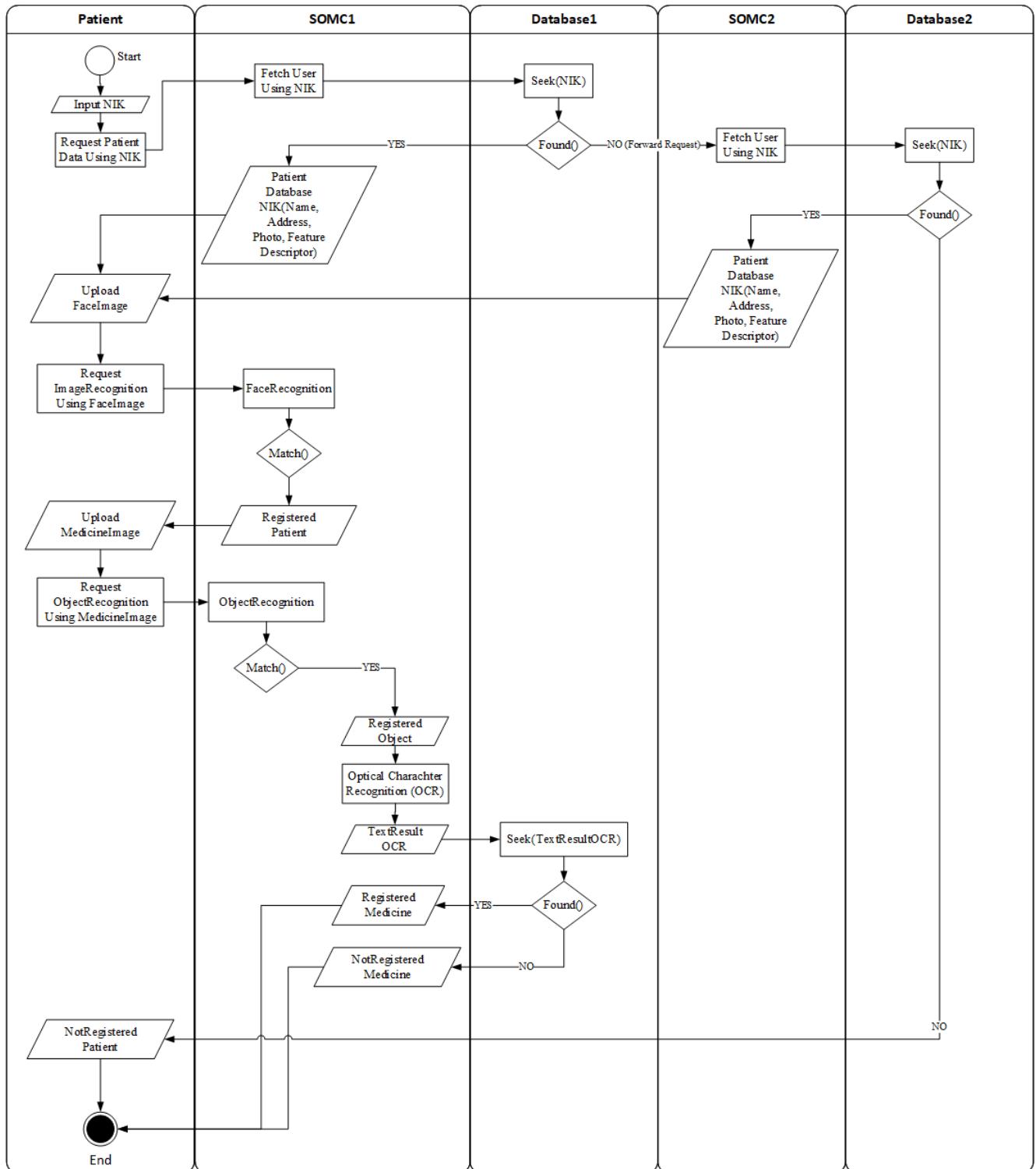


Fig. 3. ASAM activity diagram

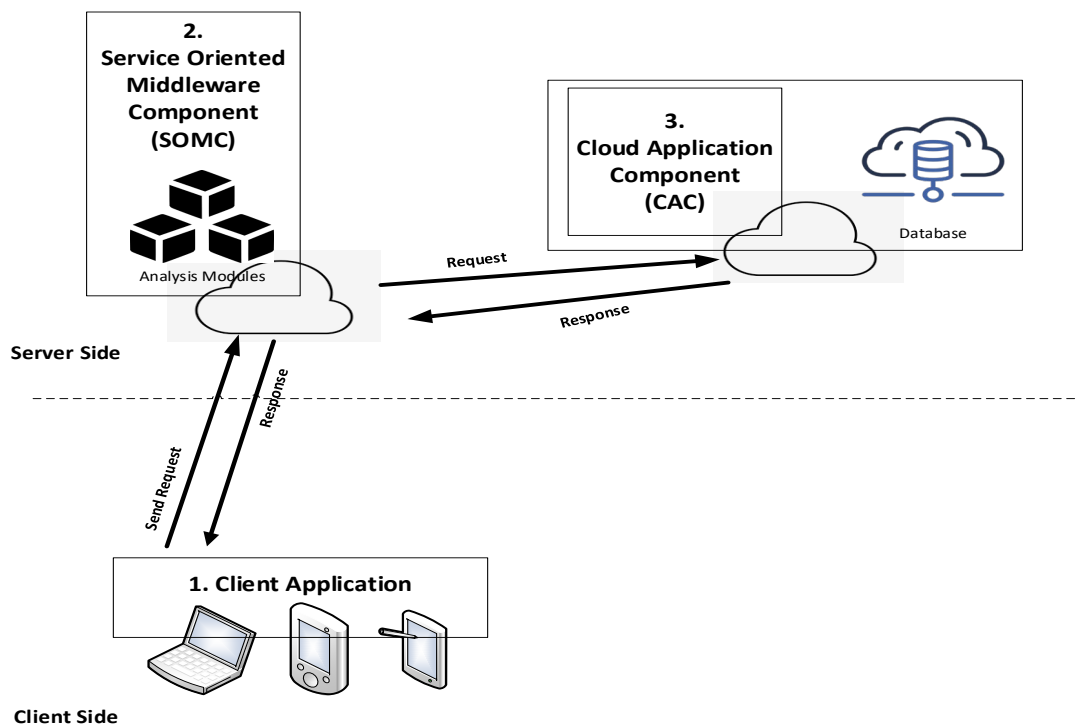


Fig. 4. ASAM architecture

for client applications. Some of the components in the middleware shown in Fig. 4 and the explanations is as follows: (1) Data provision service format and attributes is in the service builder section. (2) Data processing functions are methods that are built according to the needs of the service builder component. This component is served in a controller class. (3) Analysis modules are a module containing business rule functions based on TB's information management from health centers, which are supported by analysis functions from the machine learning models that have been created. The format and attributes of this component are managed in the helper class. (4) The data manipulation command function is built based on the rules of the controller class and helper class. This function is built in the form of a method in the model class. (5) The communication command function with the database server is managed at the connector layer. This function is built based on the rules of the library module of each type and the database server platform, then implemented in the connector class.

After SOMC is built, this Middleware can perform several service functions as follows: First, it receives requests from client applications according to the format and attributes provided. Next, execute forward requests to cloud services. Further, build communication with CAC if needed based on request. Finally, perform data processing according to request and send response to application client. CAC as a cloud service, is a cloud application provided for SOMC and application clients. CAC components can be in the form of web services, database servers, library modules, or analysis modules (Balasubramanian et al., 2020).

Analysis modules can be built by ourselves or obtained from a machine learning provider company, such as teachable machines with Google and AWS.

Moreover, The ASAM Management Component based on REST Web Service shown in Fig. 5 is principally an extension of the previously developed SOM architecture to provide data exchange services (Sathis Kumar and Latha, 2020; Suprihadi et al., 2020). The development of the SOM architecture is the addition of an Analysis Builder layer. This layer is intended to provide a data analysis module, so that the ASAM model can guarantee a SOM architecture that has a data analysis layer that can be expanded and provisioned as needed. The Analysis Builder layer is guaranteed to be implemented because in principle it has been provided by several programming language frameworks, one of which is the CI framework which is part of the Helper Class.

In this research, the Analysis Builder layers include the addition of three analysis modules, including face recognition, object recognition, and optical character recognition (OCR). The three analysis modules were chosen because they were considered in this study as tools and proposed mechanisms for identifying patients taking medication, in supporting TB elimination programs in the treatment of TB patients by presenting evidence-based information. The face recognition module uses the open source faceDetection.js and faceRecognition.js modules.

The object recognition module uses the teachable machine service provided by Google. Meanwhile, the OCR module uses opticalCharacterRecognition.js which is also open source. This paper does not discuss the workings and

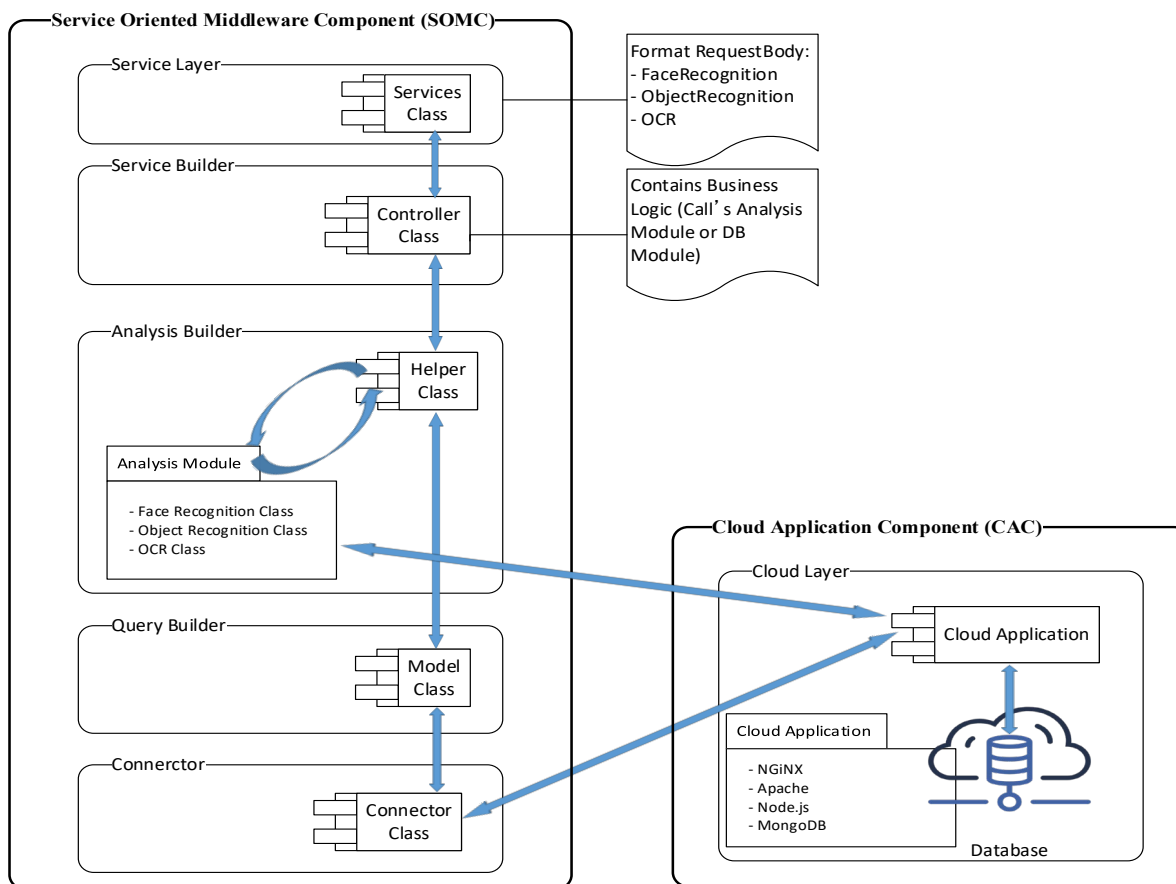


Fig. 5. ASAM management component based on REST web service

performance of the three analysis modules, but what is discussed is how to implement, how to deploy, and measure performance as a component of the ASAM architecture.

3. RESULTS AND DISCUSSION

3.1 ASAM: Implementation

ASAM is essentially a middleware application. In this paper the implementation of ASAM uses the JavaScript programming language, then is placed on the Node.js web server environment platform. The database used is mongoDB and is allocated a different address than the ASAM application. To clarify the ASAM implementation offered in the TB elimination program, it can be seen in Fig. 6 of our system architecture. Every health facility is required to implement ASAM in the patient service information system application, so that it can provide services to a patient even though the patient's data is not stored in its database. Thus, the problem of isolating the patient's domicile due to economic and social problems can be resolved, because patients can be served at every health facility. Each service request requires the patient's NIK data, namely the ID Number from Resident Card that must be owned by Indonesian citizens, and the ID Number is unique.

The proposed ASAM model is implemented on three web

hosting servers where the service addresses and specifications can be seen in Table 1. The choice of hosting server specifications with 1 virtual CPU (VCPU) with 1 GB of internal memory is the minimum specification required for implementing the ASAM model. Then, a hosting server specification is also chosen that can provide up to 8 VCPU. This is to prove that there is a correlation between the quality of the ASAM model's service performance and the specifications of the server computer used.

For service requests, HTTP REQUEST uses the GET and POST methods with the calling format as shown in Table 2. Both methods are used for patient recognition services, namely the GET method based on NIK, while the POST method is based on the results of the face recognition analysis module. For drug recognition, use the POST method to call analysis module services in the form of object recognition and OCR. Table 2 shows the implementation of the ASAM model on the Digital Ocean Server, while the other two servers have identical parameters, data resources, and structures.

To clarify the implementation of the ASAM model-based service, which in principle is a web service application with the http protocol, this paper uses the JMeter application for execution of request and response services. Fig. 7 shows one of the service request executions in the form of face recognition on a hosting server, which calls the patient.js

module as a service class. Patient.js source code can be seen in Source Code 1. Meanwhile, for other Service Classes that are in the Service Layer of the ASAM model, namely face.js, object.js, and OCR.js as shown in Fig. 8.

Service Class is a module whose job is to manage the business role of an analytic function. In this paper, patient.js is tasked with managing evidence-based patient validity

analysis, namely NIK and face recognition. Initial data was obtained from patient registration, including NIK and facial photos as patient image data. The patient's image data is then stored in the TB's database in the form of a face descriptor. Thus, the patient's validity analysis was carried out based on NIK and the patient's facial image as shown in Algorithm 1.

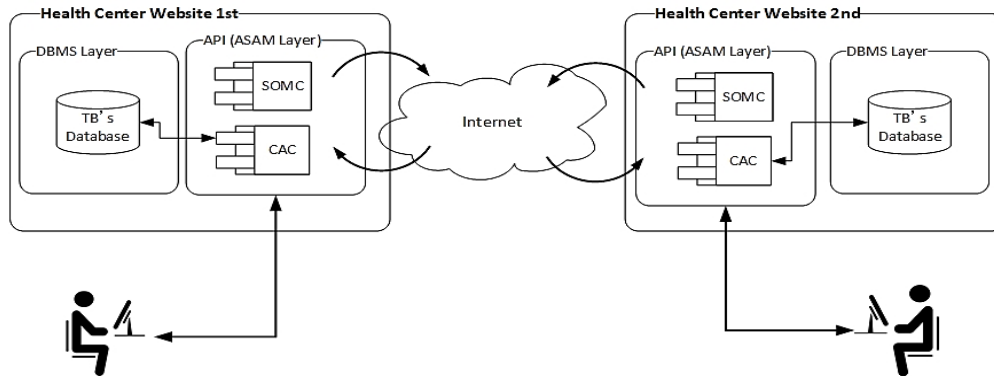


Fig. 6. System architecture

Table 1. List of web services and server hosting specifications

Server name	Virtual CPU (up to)	RAM (up to)	Service URL
Digital Ocean	1	1 GB	https://recognition-app-az6uk.ondigitalocean.app
Railway	8	800 MB	https://recognition-app-production.up.railway.app
Render	1	1 GB	https://recognition-app.onrender.com

Table 2. List of services for ASAM model on digital ocean server

Services name	Service URL	Method	Path	Image size
Get by NIK other server	https://recognition-app-az6uk.ondigitalocean.app	GET	/pasien/?nik	-
Face recognition	https://recognition-app-az6uk.ondigitalocean.app	POST	/pasien/?nik/recognize	30.2 kb
Object recognition	https://recognition-app-az6uk.ondigitalocean.app	POST	object	4.32 kb
OCR	https://recognition-app-az6uk.ondigitalocean.app	POST	ocr	4.32 kb

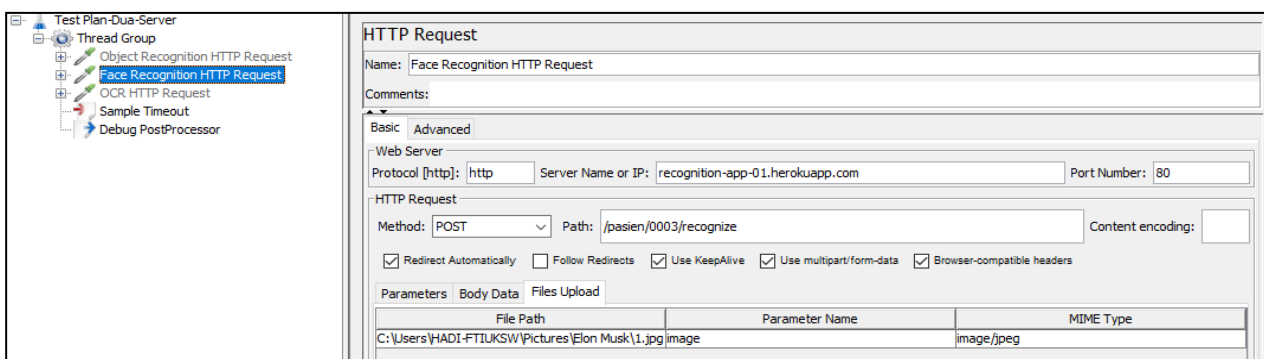


Fig. 7. Request service

face.js	initial commit	6 months ago
object.js	feat: update helper	last month
ocr.js	initial commit	6 months ago
pasien.js	feat: implementation of mongodb	5 months ago

Fig. 8. Implementation service layer

Algorithm 1: Service class for patient validity

```

1: Initialization: //Get library.
2: const response           = Fetch library: response
3: const ResponseCode       = Fetch library:
  responseCode
4: const PasienRepository   = Fetch library: pasien
5: const GetFaceDescriptor  = Fetch library:
  faceDetection
6: const CompareDecriptor   = Fetch library:
  faceRecognition
7: function GetByNik(req, res, next)
8:   params                 = req.params // assign NIK to var
  params
9:   result =
  PasienRepository.getPasienAsync(params.nik) //
  fetch nik data from database
10:  GetbyNik                = response (res, result) // return to
  function
11: End function
12: function Recognize (req, res, next)
13:  queryImage = req.file // assign file image to var
  params
14:  if (queryImage == null)
15:    Recognize = response (res, null,
  ResponseCode.BadRequest) // return to function
16:  else
17:    if (queryImage != null)
18:      params = req.params // assign NIK to var
  params
19:      pasien =
  PasienRepository.getPasienAsync(params.nik) //
  fetch nik data from database
20:      source = ToLabeledFaceDescriptors(pasien.foto,
  params.nik) // fetch image data from database and
  get descriptors
21:      faceDescriptor =
  GetFaceDescriptors(queryImage.buffer) // fetch
  descriptors from request
22:      result = CompareDecriptor(source,
  faceDescriptor) // get value from function compare
23:      Recognize      = response (res, result) // return
  to function
24:    end if
25:  end if
26: end function

```

Based on Algorithm 1, lines 7–11 are the process of searching and matching NIK data between the NIK sent in the 8th line request service, and the NIK registered in the TB's database belonging to the 9th line health facility. Lines 12–26 are the process of searching and matching face descriptor data based on registered NIK. If the two data matching processes produce True response data, then the next process will proceed, namely the analysis of the validity of TB Medicine. In lines 5 and 6 it proves that analysis modules obtained from Open-Source providers in the form of machine learning and AI applications can be

used in this ASAM-based middleware application which is placed in the Helper folder. In this paper, four analysis modules based on Open Source are used which are placed in the Helper folder as shown in Fig. 9, including faceDetection.js, faceRecognition.js, objectRecognition.js, and opticalCharacterRecognition.js.

3.2 Discussions

To justify the feasibility of the proposed ASAM model using the characteristic aspects of a web service-based middleware application. One approach is Message Oriented Middleware (MOM), including coupling, reliability, scalability, and availability (Mahmoud, 2005). The ASAM model is principally a SOM developed from service oriented architecture (SOA) technology, where the feasibility characteristics of SOA have developed into eight aspects, including functional suitability, performance efficiency, compatibility, usability, reliability, security, maintainability, and portability (França and Soares, 2015). In this paper, testing the feasibility of the ASAM model uses only three characteristic aspects, including functionality, efficiency, and reliability because it focuses on the feasibility of a web service application (Tjung et al., 2020).

The calculation of the functionality and reliability aspects uses a formula obtained from ISO/IEC 9126, namely the calculation of the functionality aspect as shown in Equation (1), while the reliability aspect is as shown in Equation (2) (Pusparani et al., 2023). For the efficiency aspect, in principle, performance testing is measured based on the total time used to process requests and response services for a web service. These three aspects were tested using the Apache JMeter tools and the results are displayed in the form of a summary report table and response time graphs.

$$F = 1 - \frac{\Sigma_{\text{failed}}}{\Sigma_{\text{success}}} \quad (1)$$

Note that F is functionality aspect. Moreover Σ_{failed} is a totally failed functionality test case and Σ_{success} is total success functionality test cases.

$$R = 1 - \frac{\Sigma_{\text{failed}}}{\Sigma_{\text{executed}}} \quad (2)$$

Note that R is *reliability* aspect. Σ_{failure} is the total number of failures that were detected during a defined trial period. Σ_{executed} is the number of executed test cases. The ASAM model proposed in this paper is a middleware application using a REST web service. Therefore, the black box testing method scenario is used for 50 sample request services that are implemented in the JMeter tools application, to test the feasibility of implementing the proposed ASAM model. The test scenarios are grouped based on the type of service provided by the ASAM model proposed in this paper as presented in Table 2. This test uses internet services with standard speeds that are owned on the 4G LTE network.

Furthermore, we evaluate our proposed method with 4 different services including get by NIK other server service, Face Recognition Service, Object Recognition Service, and Face Recognition Service.

The Get by NIK other server service is a service for searching patient NIK data stored in a central database. This means that the patient's NIK data entered is not stored in the local server database or the patient data is not on a server computer that stores the ASAM application that the user accesses. The test results with 50 test cases can be seen in Table 3 Summary Report for Get by NIK other Server Service. Based on Table 3, it shows that the functionality of the ASAM is 1. This means that the ASAM model can function properly without errors in providing data search services.

Face recognition service is an example of a web service that is capable of processing analysis modules in the form of face recognition. In the proposed ASAM model, the face recognition service uses the POST method to execute service requests. The data sent is text data, namely NIK, and 30.2 kb JPG format image data. The image data is sent to the server and then converted into a data descriptor format to be processed by the face recognition module. The response given is true or false, where true means the image is recognized as a registered patient, while false means the

patient is unknown.

With several stages of the process, the ASAM model tested as shown in Table 4 shows that it has a good aspect of functionality ($F = 1$), which means the service is run without any errors. According to Table 4, this test demonstrates that the analysis service that was developed on the ASAM model has a solid efficiency component because it has a response time that is on average less than four seconds across three separate servers.

Furthermore, the proposed ASAM model tries to implement an analysis module in the form of object recognition, which is a form of machine learning provided by the TensorFlow library. The analysis module in this paper is intended to function as a form recognition tool for a drug. The data sent is in the form of image data containing an example of a drug image measuring 4.32 Kb, using the POST method. The image data is processed by the object recognition module on the server side which has the ASAM model application. The response given is a true value which means the drug is recognized, and a false value means the drug is not known.

Nevertheless, the results of testing the object recognition service with 50 sample request services are presented in Table 5. The three different servers running 50 samples of ASAM model-based object recognition services have been

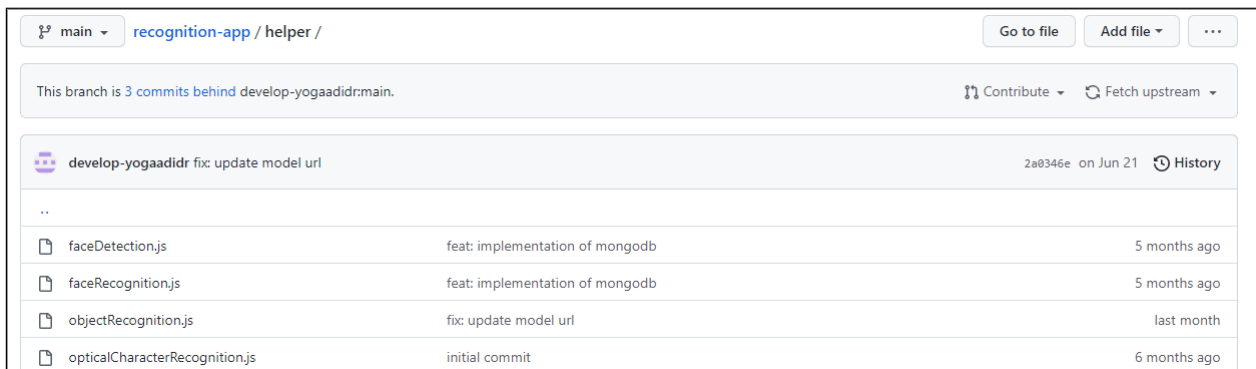


Fig. 9. List of analysis module

Table 3. Summary report for get by NIK other server service

Service name (Server name)	# Samples (a)	Average time (ms) (b)	Min time (ms) (c)	Max time (ms) (d)	Error % (e)	$\frac{\Sigma_{\text{failed}}}{\Sigma_{\text{success}}} = \frac{(e)}{(a)}$ (f)	$F = 1 - (f)$ (g)
Get NIK Server 1 (Digital Ocean)	50	2223	1873	7963	0.00%	0	1
Get NIK Server 2 (Railway)	50	1792	1594	2029	0.00%	0	1
Get NIK Server 3 (Render)	50	2073	1626	2550	0.00%	0	1

Table 4. Summary report for face recognition service

Service name (Server name)	# Samples (a)	Average time (ms) (b)	Min time (ms) (c)	Max time (ms) (d)	Error % (e)	$\frac{\Sigma_{\text{failed}}}{\Sigma_{\text{success}}} = \frac{(e)}{(a)}$ (f)	$F = 1 - (f)$ (g)
Face recognition Server 1 (Digital Ocean)	50	2730	2350	4637	0.00%	0	1
Face recognition Server 2 (Railway)	50	2127	1948	2937	0.00%	0	1
Face recognition Server 3 (Render)	50	3103	2592	4012	0.00%	0	1

shown to run without any errors. Likewise, the service is run without any failure in providing a response. The sample data from the first to the fiftieth on each server gives a response code of 200 which means it is valid. Response time for this object recognition service takes an average of less than 2 seconds. In other words, the proposed ASAM model has aspects of functionality, reliability, and efficiency that are appropriate for analysis services in the form of object recognition.

Next, OCR service is a web service that provides analysis results from an AI module, to recognize writing in the form of a drug name printed on the drug or drug cover. This OCR analysis module, applied to the proposed ASAM model and tested with 50 service samples, aims to assess its feasibility as a SOM model.

Table 6 contains a summary report for OCR service on three different servers. The result shows that the ASAM model has good functionality and reliability aspects. This is indicated by an error value of 0 percent without any failure response. In terms of efficiency, the ASAM model has a minimum performance, which has an average response time of more than 4 seconds on the first and third servers. This is

because the two servers only have one virtual CPU or core. Whereas the second server has a good response time because the server provides processor specifications of up to 8 virtual CPUs.

Moreover, Fig. 10 describes the efficiency or reliability aspect of the proposed ASAM model and is very clearly influenced by the specifications of the processor, not memory (RAM). For testing the reliability aspect for the Get by NIK other Server Service shown in Fig. 10(a), it shows that out of 50 ASAM model request services can provide valid response data as specified. Meanwhile, the efficiency aspect of testing on three different servers shows that the ASAM model implemented takes an average time of under 4 seconds. Fig. 10(b) illustrates that after 50 iterations of reliability testing, we may conclude that the request services for the ASAM model can produce correct return data as stated. Meanwhile, testing conducted across three servers reveals that the ASAM model in use typically completes a transaction in less than four seconds, making it highly efficient. This proves that the ASAM model has a decent performance as a web service application.

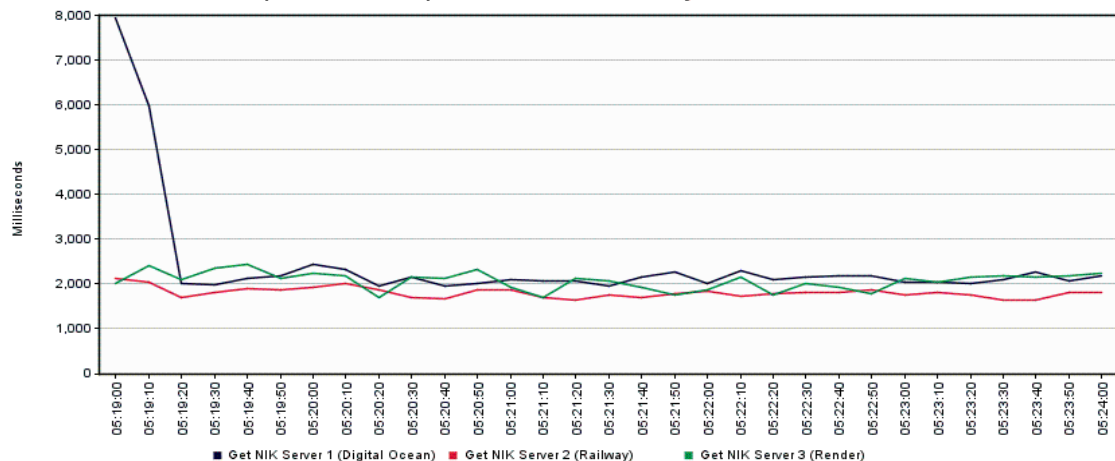
Table 5. Summary report for object recognition service

Service name (Server name)	# Samples	Average time	Min time	Max time	Error %	$\frac{\Sigma_{\text{failed}}}{\Sigma_{\text{success}}} = \frac{(e)}{(a)}$	$F = 1 - (f)$
		(ms)	(ms)	(ms)		(f)	
	(a)	(b)	(c)	(d)	(e)	(f)	(g)
Object detection Server 1 (Digital Ocean)	50	1466	1351	1798	0.00%	0	1
Object detection Server 2 (Railway)	50	1792	1732	1958	0.00%	0	1
Object detection Server 3 (Render)	50	1956	1547	2258	0.00%	0	1

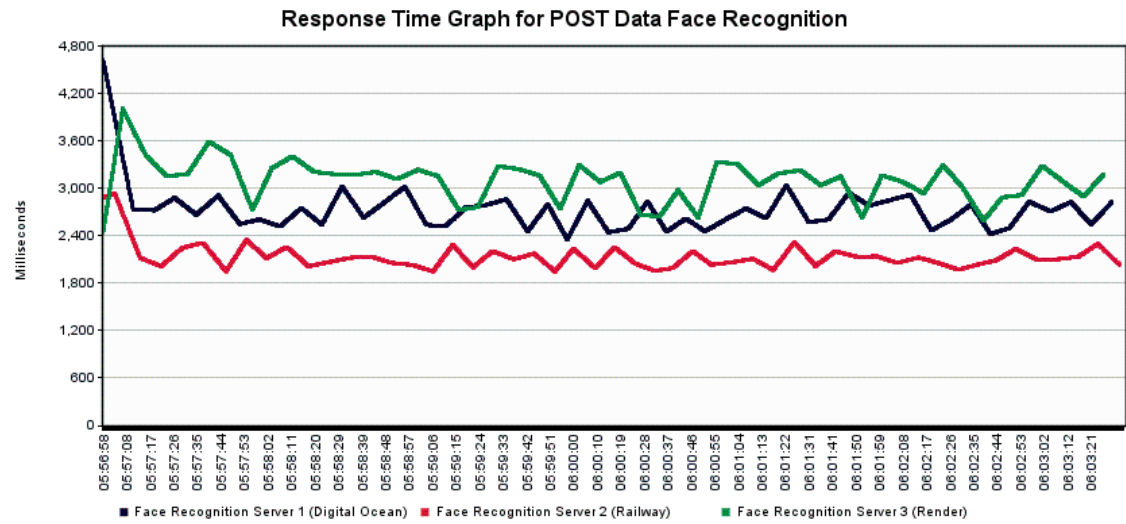
Table 6. Summary report for OCR service

Service name (Server name)	# Samples	Average time	Min time	Max time	Error %	$\frac{\Sigma_{\text{failed}}}{\Sigma_{\text{success}}} = \frac{(e)}{(a)}$	$F = 1 - (f)$
		(ms)	(ms)	(ms)		(f)	
	(a)	(b)	(c)	(d)	(e)	(f)	(g)
OCR Server 1 (Digital Ocean)	50	5587	5266	7503	0.00%	0	1
OCR Server 2 (Railway)	50	1725	1624	2123	0.00%	0	1
OCR Server 3 (Render)	50	6339	5044	8686	0.00%	0	1

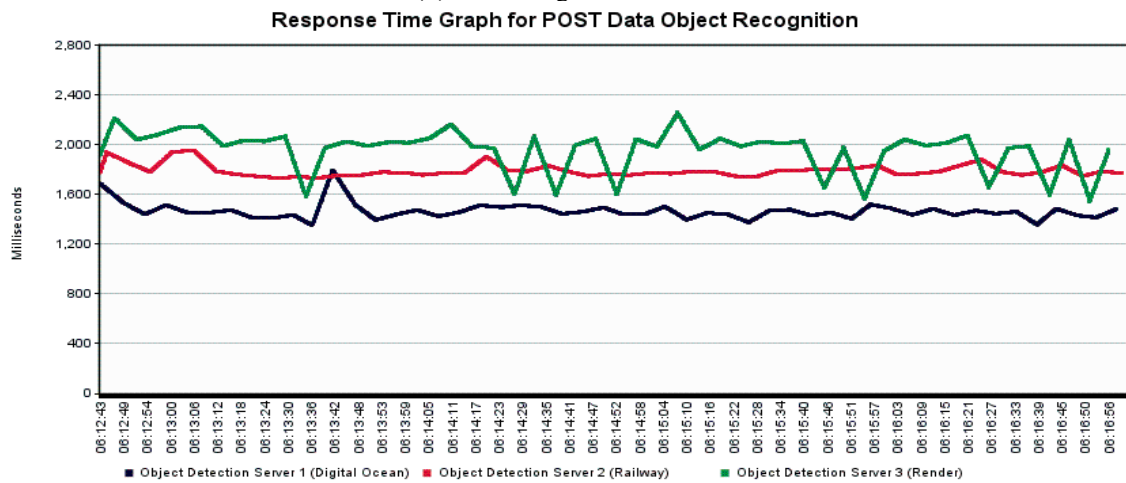
Response Time Graph for GET Data Patient by NIK at Other Server



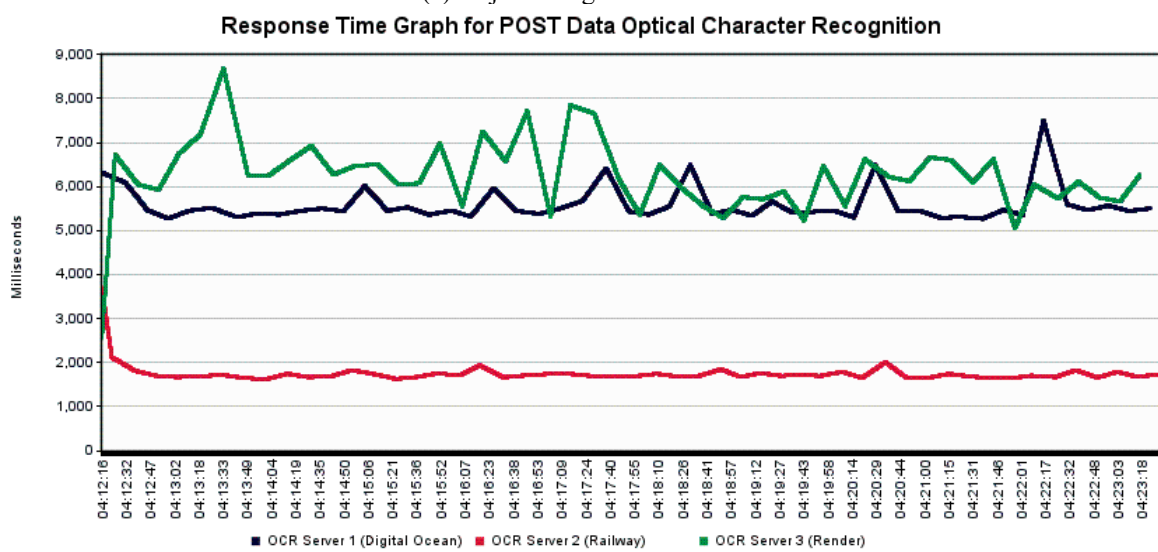
(a) Get by NIK other server service



(b) Face recognition service



(c) Object recognition service



(d) OCR service

Fig. 10. Response time graph for (a) Get by NIK other server service, (b) Face recognition service, (c) Object recognition service, and (d) OCR service

4. CONCLUSION

In this article, we provide the ASAM model is a novelty in SOM, as an application of Service Oriented Architecture technology for exchanging messages and providing data analysis services. ASAM can be used one to develop data and information integration systems between stakeholders, especially health facilities that have the same business processes by prioritizing data autonomy. With ASAM, the TB epidemic elimination program in Indonesia can be realized, with the development of middleware applications to assist evidence-based patient medication adherence services. Thus, the management of TB information between stakeholders of the national TB elimination program, can be developed cheaply and independently by the Indonesian government. The further research needed is the development of an ASAM-based TB patient care service system, and measuring the success rate of the impact of using this IT in TB elimination programs. Moreover, we evaluate our proposed method with 4 different services including get by NIK other server service, Face Recognition Service, Object Recognition Service, and Face Recognition Service. Based on the evaluation results shows that the functionality of the ASAM model is worth 1. This means that the ASAM model can function properly without errors in providing data search services.

In the future, we want to invest in building better methods that eliminate the need to retry fetching a response from the middleware when a time-out occurs in a computer-intensive service. The proposed method is mainly concerned with minimizing the waiting time of mobile users to access resources and continue to enjoy uninterrupted use of resource-intensive services. They will run tests to see how well they perform in conditions where heavy service usage is likely to exceed mobile device timeout limits.

ACKNOWLEDGMENT

The authors would like to thank the support and help from, Satya Wacana Christian University, Central Java provincial government, in this case the Balkesmas in the Magelang region, for supporting the availability of this research data and others that took part in this work.

REFERENCES

- Abdelfattah, A.S., Abdelkader, T., EI-Horbaty, E.S.M. 2020. RAMWS: Reliable approach using middleware and WebSockets in mobile cloud computing. *Ain Shams Engineering Journal*, 11, 1083–1092.
- Abdelfattah, A.S., Abdelkader, T., EI-Horbaty, E.S.M. 2018. RSAM: An enhanced architecture for achieving web services reliability in mobile cloud computing. *Journal of King Saud University-Computer and Information Sciences*, 30, 164–174.
- Alshamrani, M. 2022. IoT and artificial intelligence implementations for remote healthcare monitoring systems: A survey. *Journal of King Saud University-Computer and Information Sciences*, 34, 4687–4701.
- Ardhianto, P., Liau, B.Y., Jan, Y.K., Tsai, J.Y., Akhyar, F., Lin, C.Y., Subiako R.B.R., Lung, C.W. 2022. Deep learning in left and right footprint image detection based on plantar pressure. *Applied Sciences*, 12, 8885.
- Bajaj, K., Jain, S., Singh, R. 2023. Context-aware offloading for iot application using fog-cloud computing. *International Journal of Electrical and Electronics Research*, 11, 69–83.
- Balasubramanian, N., Gurumurthy, T.R., Bharat, D. 2020. Receiver based contention management: A cross layer approach to enhance performance of wireless networks. *Journal of King Saud University-Computer and Information Sciences*, 32, 1117–1126.
- Calimeri, F., Caeteruccio, F., Cinelli, L.U.C.A., Marzullo, A., Stamile, C., Terracina, G., Durand-Dubief, F., Sappey-Marinier, D. 2019. A logic-based framework leveraging neural networks for studying the evolution of neurological disorders. *Theory and Practice of Logic Programming*, 21, 80–124.
- Cao, L. 2022. AI in finance: Challenges, techniques, and opportunities. *ACM Computing Surveys*, 55.
- Castelli, M., Manzoni, L., Espindola, T., Popovič, A., De Lorenzo, A. 2021. Generative adversarial networks for generating synthetic features for Wi-Fi signal quality. *PLoS ONE*, 16, e0260308.
- Champaneria, T., Jardosh, S., Makwana, A. 2022. Microservices in IoT middleware architectures: Architecture, trends, and challenges. *IOT with Smart Systems: Proceedings of ICTIS 2022*, 2, 381–395.
- Chen, Y., Lin, C., Huang, J., Xiang, X., Shen, X. 2017. Energy efficient scheduling and management for large-scale services computing systems. *IEEE Transactions on Services Computing*, 10, 217–230.
- Dewi, C., Chen, A.P.S., Christanto, H.J. 2023a. Deep learning for highly accurate hand recognition based on Yolov7 model. *Big Data and Cognitive Computing*, 7, 53.
- Dewi, C., Chen, A.P.S., Christanto, H.J. 2023b. Recognizing Similar musical instruments with YOLO models. *Big Data and Cognitive Computing*, 7, 94.
- Dewi, C. Chen, R.-C. 2022. Automatic medical face mask detection based on cross-stage partial network to combat COVID-19. *Big Data and Cognitive Computing*, 6, 106.
- Dewi, C., Chen, R.-C., Zhuang, Y.-C., Jiang, X., Yu, H. 2023c. Recognizing road surface traffic signs based on Yolo models considering image flips. *Big Data and Cognitive Computing*, 7, 54.
- Dewi, C., Chen, R.C., Yu, H., Jiang, X. 2021. Robust detection method for improving small traffic sign recognition based on spatial pyramid pooling. *Journal of Ambient Intelligence and Humanized Computing*, 12, 1–18.
- Dewi, C., Chen, A.P.S., Christanto, H.J. 2023d. YOLOv7 for face mask identification based on deep learning. 2023 15th International Conference on Computer and

- Automation Engineering (ICCAE), 193–197.
- Falzon, D., Migliori, G.B., Jaramillo, E., Weyer, K., Joos, G., Raviglione, M. 2017. Digital health to end tuberculosis in the Sustainable Development Goals era: Achievements, evidence and future perspectives. *European Respiratory Journal*, 50.
- Findi, M. 2021. Aplikasi Digital dalam Mendukung Layanan Kesehatan untuk Eliminasi Tuberkulosis di Indonesia.
- França, J.M., Soares, M.S. 2015. SOAQM: Quality model for SOA Applications based on ISO 25010. In *ICEIS*, 60–70.
- Gamal, I., Abdel-Galil, H. Ghalwash, A. 2022. Osmotic message-oriented middleware for Internet of Things. *Computers*, 11, 56.
- Giao, J., Nazarenko, A. A., Luis-Ferreira, F., Gonçalves, D., Sarraipa, J. 2022. A framework for service-oriented architecture (SOA)-based IoT application development. *Processes*, 10, 1782.
- Huang, J., Lin, C., Kong, X., Wei, B., Shen, X. 2014. Modeling and analysis of dependability attributes for services computing systems. *IEEE Transactions on Services Computing*, 7, 599–613.
- Huang, J., Lin, C., Wan, J. 2013. Modeling, analysis and optimization of dependability-aware energy efficiency in services computing systems. *Proceedings - IEEE 10th International Conference on Services Computing, SCC*, 683–690.
- Kuhn Cuellar, L., Friedrich, A., Gabernet, G., de la Garza, L., Fillinger, S., Seyboldt, A., Koch, T., zur Oven-Krockhaus, S., Wanke, F., Richter, S., Thaiss, W.M., Horger, M., Malek, N., Harter, K., Bitzer, M., Nahnsen, S. 2022. A data management infrastructure for the integration of imaging and omics data in life sciences. *BMC Bioinformatics*, 23, 61.
- Lamnaour, M., Begdouri, M.A., Mesmoudi, Y., El Khamlichi, Y., Tahiri, A. 2022. A semantic MSOAH-IoT design for improving efficiency and solving heterogeneity within IoT applications. *Journal of Communications*, 17.
- Lee, Y., Raviglione, M.C., Flahault, A. 2020. Use of digital technology to enhance tuberculosis control: Scoping review. *Journal of Medical Internet Research*, 22, 1–15.
- Lyu, G., Liu, P., Lu, Y., Wang, T., Kang, X. 2021. A data middle platform architecture based on microservice serving power grid business. 2021 8th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)/2021 7th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom). 219–224.
- Mahmoud, Q.H. 2004. *Middleware for communications*, 73. John Wiley & Sons.
- Mesmoudi, Y., Lamnaour, M., El Khamlichi, Y., Tahiri, A., Touhafi, A., Braeken, A. 2020. A middleware based on service oriented architecture for heterogeneity issues within the Internet of Things (MSOAH-IoT). *Journal of King Saud University-Computer and Information Sciences*, 32, 1108–1116.
- Mohamed, N., Al-Jaroodi, J., Jawhar, I., Lazarova-Molnar, S., Mahmoud, S. 2017. SmartCityWare: A service-oriented middleware for cloud and fog enabled smart city services. *IEEE Access*, 5, 17576–17588.
- Naseer, A., Alkazemi, B.Y., Aldoobi, H.I. 2016. A general-purpose service-oriented middleware model for WSN. *International Conference on Ubiquitous and Future Networks, ICUFN*, 283–287.
- Nayak, N., Ambalavanan, U., Thampan, J.M., Grewe, D. 2023. Reimagining automotive service-oriented communication: A case study on programmable data planes. *IEEE Vehicular Technology Magazine*, 2, 2–12.
- Ngwatu, B.K., Nsengiyumva, N.P., Oxlade, O., Mappin-Kasirer, B., Nguyen, N.L., Jaramillo, E., Falzon, D., Schwartzman, K. 2018. The impact of digital health technologies on tuberculosis treatment: A systematic review. *European Respiratory Journal*, 51.
- Phuttharak, J., Loke, S.W. 2023. An event-driven architectural model for integrating heterogeneous data and developing smart city applications. *Journal of Sensor and Actuator Networks*, 12, 12.
- Prisilla, A.A., Guo, Y.L., Jan, Y.K., Lin, C.Y., Lin, F.Y., Liao, B.Y., Tsai, J.Y., Ardhiyanto, P., Pusparani, Y., Lung, C.W. 2023. An approach to the diagnosis of lumbar disc herniation using deep learning models. *Frontiers in Bioengineering and Biotechnology*, 11.
- Pusparani, Y., Lin, C.Y., Jan, Y.K., Lin, F.Y., Liao, B.Y., Ardhiyanto, P., Farady, I., Alex, J.S.R., Aparajeeta, J., Chao, W.H., Lung, C.W. 2023. Diagnosis of Alzheimer's disease using convolutional neural network with select slices by landmark on hippocampus in MRI images. *IEEE Access*, 11.
- Riono, P. 2019. Tantangan Kita Mencapai Eliminasi Tuberkulosis di Indonesia tahun 2030. *Jurnal Kesehatan*: 2–9.
- Rohmah, R.N., Handaga, B., Nurokhim, N., Soesanti, I. 2019. A statistical approach on pulmonary tuberculosis detection system based on X-ray image. *Telkomnika (Telecommunication Computing Electronics and Control)*, 17, 1474–1482.
- Sathis Kumar, T., Latha, K. 2020. Middleware interoperability performance using interoperable reinforcement learning technique for enterprise business applications. *Journal of Intelligent & Fuzzy Systems*, 38.
- Sembiring, J., Uluwiyah, A. 2015. Data exchange design with SDMX format for interoperability statistical data. *TELKOMNIKA Indonesian Journal of Electrical Engineering*, 14, 343–352.
- Sharif, Z., Jung, L.T., Ayaz, M., Yahya, M., Pitafi, S. 2023. Priority-based task scheduling and resource allocation in edge computing for health monitoring system. *Journal of King Saud University-Computer and Information Sciences*, 35, 544–559.
- Suhardi, N.K., Sembiring, J. 2017. Service computing system engineering life cycle. *Proceeding of the Electrical Engineering Computer Science and*

- Informatics, 4, 347–352.
- Sukarsa, I., Wisswani, N.W., Wirabuana, P. 2014. Data exchange between information system at low bandwidth quality using messaging. *Journal of Theoretical and Applied Information Technology*, 60, 417–422.
- Suprihadi, S., Wijono, S., Hartomo, K.D. 2020. Service oriented middleware for tuberculosis information services management. 2020 International Seminar on Application for Technology of Information and Communication (iSemantic), 425–430.
- Teixeira, T., Hachem, S., Issarny, V., Georgantas, N. 2011. Service oriented middleware for the internet of things: A perspective. In *European conference on a service-based internet*, 220–229.
- Tjung, Y., Wibowo, A., Ridha, M.A.F. 2020. A model and implementation of academic data integration in near-real time using message-oriented middleware to support analysis of student performance in the information technology department of Politeknik Caltex Riau. *JUTI: Jurnal Ilmiah Teknologi Informasi*, 18, 9–18
- Wang, N., Zhang, H., Zhou, Y., Jiang, H., Dai, B., Sun, M., Li, Y., Kinter, A., Huang, F. 2019. Using electronic medication monitoring to guide differential management of tuberculosis patients at the community level in China. *BMC Infectious Diseases*, 19, 1–9.
- Wisswani, N.W., Wijaya, I.W.K. 2018. Message oriented middleware for library's metadata exchange. *Telkomnika (Telecommunication Computing Electronics and Control)*, 16, 2756–2762.
- World Health Organization. 2017. Handbook for the use of digital technologies to support tuberculosis medication adherence.
- World Health Organization. 2020. Global tuberculosis report 2020. World Health Organization, Geneva.
- World Health Organization. 2021. Global tuberculosis report 2021. Geneva.
- Wu, Z. 2014. *Service computing: Concept, method and technology*. Academic Press.
- Ye, D., He, Q., Wang, Y., Yang, Y. 2019. An Agent-based integrated self-evolving service composition approach in networked environments. *IEEE Transactions on Services Computing*, 12, 880–895.
- Zhang, Y., Tang, D., Zhu, H., Zhou, S., Zhao, Z. 2022. An efficient IIoT gateway for cloud-edge collaboration in cloud manufacturing. *Machines*, 10.