RIK International Journal of Science and Technological Research (RIK-IJSTR) *Vol. 7 No. 1, January 2021*
ISSN: 2350-6310 (Print)
ISSN: 2350-543X (Online)

An Improved Model for Service-Oriented Web-Architecture using Swarm Intelligence Technique

Kingsley-Opara, Ngozi
Research Scholar
Department of Computer Science
Ignatius Ajuru University of Education
Rivers State
Nigeria

&

Onuodu, Friday Eleonu
Visiting Scholar
Department of Computer Science
University of Port-Harcourt
Rivers State
Nigeria

ABSTRACT

A Service Oriented Architecture (SOA) is a paradigm that focuses on building a system through the use of different reusable web services. This technique ensures the reusability of software components or services by other applications during software development thereby simplifying the whole process and reducing the development cost. This has made web service delivery efficient in many ways. However, there are still issues of incompatibility and lack of action procedure for effective service response. These led to a need for a better service-oriented model. In this work, we proposed an improved model for service oriented web architecture using swarm intelligence technique. Swarm intelligence technique is a technology that is modelled after biological way of harmonious community behaviour among some organisms like bees, ant, and fish. The Object-oriented Design Methodology (OODM) was used in this approach. The system was implemented using Hypertext Pre-processor (PHP) and JavaScript as the programming languages and MySQL as Database. Result from the system testing showed high performance accuracy in speed of service request response and execution of action procedure. Based on these parameters, our system recorded a performance accuracy score of 86%, while the existing system scored 66%. This work could be beneficial to software developers on the web, to web service consumers and to researchers in the field of artificial intelligence especially swarm intelligence.

Keywords: Web Architecture, Swarm Intelligence, Service-Oriented Model, Artificial Intelligence.

Introduction

Nature has really taught us about harmonious living that brings effectiveness in service. Taking a look at these lower organisms, human being the more advanced creature has to understudy these little creatures on how they work together without any of them infringing on the path of the other. It is so interesting to see how ants form a colony that enable them build their anthills through the collaborative work of one another, the fishes of same species school together to provide protection among themselves against predators, and many others. The swarm intelligence technique therefore takes its inspiration from these biological behaviours. A swarm is a large number of homogenous, simple agents interacting locally among themselves, and their environment, with no central control to allow a global interesting behaviour to emerge. Swarm intelligence (SI) is the collective behaviour of decentralized, self-organized systems, natural or artificial. SI systems are typically made up of a population of simple agents interacting locally with one another and with their environment.

The services of service-oriented web-architecture can be improved using the swarm intelligence technique. Service-oriented architecture (SOA) is a software development model that allows services to communicate across different platforms and languages to form applications. In SOA, a service is a self-contained unit of software designed to complete a specific task. Service-oriented architecture allows various services to communicate using a loose coupling system to either pass data or coordinate an activity.

Statement of the Problem

The existing web-based SOAs perform efficient service delivery when all the components and services required by the users and other applications are readily available. However, in the case where the required services or components are not available, the SOAs provides no cause or action i.e. action procedure. Another challenge of the web-based SOA is the incompatibility of components in new environments when invoked. These can reduce user satisfaction of service delivery over the web platform.

Aim and Objectives

The aim of this research h is to develop an improved model for service-oriented web-architecture using swarm intelligence technique. The specific objectives are to:

a Develop a SOA that provides action procedure for unavailable user requests using swarm intelligence technique.
b Implement using Hypertext Pre-processor and JavaScript as Programming Languages and MySQL as Database.
c Compare our results with other existing system performance.

Significance of the Study

Improving customer service delivery over the web is very important and cannot be trivialized. This model will help promote customer service delivery and component

adaptability in new software environments, thereby maximizing the benefits of SOA and improving user satisfaction over the web. This work could be beneficial to software developers on the web, to web service consumers and to researchers in the field of artificial intelligence especially swarm intelligence.

Literature Review

In this section is a review of related literature swarm intelligence technique and software-oriented architecture.

Swarm Intelligence Technique

Swarm intelligence takes its root from the understanding of how some living creatures work harmoniously and accordingly. According to [1] Swarm Intelligence (SI) is an innovative distributed intelligent paradigm for solving optimization problems that originally took its inspiration from the biological examples by swarming, flocking and herding phenomena in vertebrates. Furthermore, in the need to define what swarm is, [2] explained that a swarm is a large number of homogenous, simple agents interacting locally among themselves, and their environment, with no central control to allow a global interesting behaviour to emerge. They went further to show how this is applied in artificial intelligence by stating that swam intelligence can therefore be defined as a relatively new branch of Artificial Intelligence that is used to model the collective behaviour of social swarms in nature, such as ant colonies, honey bees, and bird flocks. Still on introducing what swarm intelligence is, [3] defines it as a collective behaviour of decentralized, self-organized systems, natural or artificial. SI systems are typically made up of a population of simple agents interacting locally with one another and with their environment. The inspiration often comes from nature, especially biological systems. The agents follow very simple rules, and although there is no centralized control structure dictating how individual agents should behave, local, and to a certain degree random, interactions between such agents lead to the emergence of "intelligent" global behaviour, unknown to the individual agents. Natural examples of SI include ant colonies, bird flocking, animal herding, bacterial growth, and fish schooling.

In the understanding of big data and how swam intelligence technique can be used in the processing ability required, [4] noted that Swarm Intelligence (SI), which is based on a population of individuals, is a collection of nature-inspired searching techniques. To search a problem domain, a swarm intelligence algorithm processes a population of individuals. Each individual represents a potential solution of the problem being optimized. In swarm intelligence, an algorithm maintains and successively improves a population of potential solutions until some stopping condition is met. The solutions are initialized randomly in the search space, and are guided toward the better areas through the interaction among solutions over iterations. The swarm intelligence algorithms have shown significant achievements on solving large scale, dynamical, and multi-objective problems. With the application of the swarm intelligence, more rapid and effective methods can be designed to solve big data analytics problems. The success story of

swarm intelligence cannot be complete without making reference to the inspiration gotten from nature. To this wise, [5] asserted that inspired by nature, the algorithms used by Artificial Swarm Intelligence (ASI) systems are modelled on the biological principles of Swarm Intelligence, the phenomenon that enables flocks of birds, schools of fish, and colonies of bees to maximize their collective intellect.

Swarm Intelligence (SI) can therefore be defined as a relatively new branch of Artificial Intelligence that is used to model the collective behaviour of social swarms in nature, such as ant colonies, honey bees, and bird flocks. Although these agents (insects or swarm individuals) are relatively unsophisticated with limited capabilities on their own, they are interacting together with certain behavioural patterns to cooperatively achieve tasks necessary for their survival. The social interactions among swarm individuals can be either direct or indirect. Examples of direct interaction are through visual or audio contact, such as the waggle dance of honey bees. Indirect interaction occurs when one individual changes the environment and the other individuals respond to the new environment, such as the pheromone trails of ants that they deposit on their way to search for food sources. This indirect type of interaction is referred to as stigmergy, which essentially means communication through the environment.

Artificial Intelligence and Swarm Intelligence

Talking about the importance of artificial intelligence upon which swarm technique is drawn from, [6] in its white paper noted that Artificial Intelligence is developing fast. It will change our lives by improving healthcare (e.g. making diagnosis more precise, enabling better prevention of diseases), increasing the efficiency of farming, contributing to climate change mitigation and adaptation, improving the efficiency of production systems through predictive maintenance, increasing the security of Europeans, and in many other ways that we can only begin to imagine. At the same time, Artificial Intelligence (AI) entails a number of potential risks, such as opaque decision-making, gender-based or other kinds of discrimination, intrusion in our private lives or being used for criminal purposes.

Relevance of Swarm Intelligence in Application Domain

The relevance of swarm technique in the 21st century technology can't be over emphasised and this propelled [7] to assert that Swarm intelligence is a form of artificial intelligence (AI) inspired by the insect kingdom. In nature, it describes how honeybees migrate, how ants form perfect trails, and how birds flock. In the world of AI, swarm systems draw input from individual people or machine sensors and then use algorithms to optimize the overall performance of the group or system in real time.

In furtherance to highlighting the importance of swarm intelligence technique, [8] explained that Swarm based techniques have huge application domain covering multiple disciplines, which include power system, fuzzy system, forecasting, bio-medicine, sociological analysis, image processing, sound processing, signal processing, data analysis, process modelling, process controlling, etc. According to them, in the last two

decades numerous techniques and their variations have been developed and despite many variations are being carried out, main skeleton of these techniques remain same. With diverse application domains, most of these techniques have been modified to fit into a particular application. These changes undergo mostly in perspective of encoding scheme, parameter tuning and search strategy. Sources of real-world problems are different, but their nature sometimes found similar to other problems. Hence, swarm based techniques utilized for one of these problems can be applied to others as well.

Swarm Intelligence (SI) Models

Swarm intelligence models are referred to as computational models inspired by natural swarm systems. To date, several swarm intelligence models based on different natural swarm systems have been proposed in the literature, and successfully applied in many real-life applications. Examples of swarm intelligence models are: Ant Colony Optimization, Particle Swarm Optimization, Artificial Bee Colony, Bacterial Foraging, Cat Swarm Optimization, Artificial Immune System, and Glow worm Swarm Optimization. In this paper, we will primarily focus on two of the most popular swarm intelligences models, namely, Ant Colony Optimization and Particle Swarm Optimization. However, the area of interest as relates to the topic of discussion is shown below:

a. Ants in Nature

Since tens of millions of years ago, ants have survived different environments, climates and ages that dinosaurs, for example, did not. The secret of the remarkable ecological success of ants can be explained by a single word: sociality. Ants have demonstrated exceptional social organization in several ways: They are inclined to live in organized societies made up of individuals that cooperate, communicate, and divide daily tasks. Ants have impressive abilities in finding their way, building their nests, and locating food supplies. They are not only efficient, but hard-working and thrifty creatures that can adapt to different ecosystems and survive harsh weather conditions.

b. Particle Swarm Optimization (PSO) Model

The second example of a successful swarm intelligence model is Particle Swarm Optimization (PSO), PSO was originally used to solve non-linear continuous optimization problems, but more recently it has been used in many practical, real-life application problems. For example, PSO has been successfully applied to track dynamic systems, evolve weights and structure of neural networks, analyze human tremor, register 3D-to-3D biomedical image, control reactive power and voltage, even learning to play games and music composition. PSO draws inspiration from the sociological behaviour associated with bird flocking. It is a natural observation that birds can fly in large groups with no collision for extended long distances, making use of their effort to maintain an optimum distance between themselves and their neighbours.

c. Birds in Nature

Vision is considered as the most important sense for flock organization. The eyes of most birds are on both sides of their heads, allowing them to see objects on each side at the same time. The larger size of birds' eyes relative to other animal groups is one reason why birds have one of the most highly developed senses of vision in the animal kingdom. As a result of such large sizes of birds' eyes, as well as the way their heads and eyes are arranged, most species of birds have a wide field of view. For example, Pigeons can see 300 degrees without turning their head, and American Woodcocks have, amazingly, the full 360-degree field of view. Birds are generally attracted by food; they have impressive abilities in flocking synchronously for food searching and long-distance migration. Birds also have efficient social interaction that enables them to be capable of: flying without collision even while often changing direction suddenly, scattering and quickly regrouping when reacting to external threats, and avoiding predators.

Applications of SI

The purpose of this section is to present some real-life problems and applications in which the use of swarm-based optimization algorithms has been successfully made in the literature, but not providing readers with an extensive survey on swarm intelligence applications. More details on SI applications can, however, be found here. The impressive performance of SI algorithms in discrete and continuous optimization problems has increased the attention of many researchers with different backgrounds to apply SI algorithms into their own research areas. As a result, there has been an almost exponential increase in the number of research papers reporting the successful application of SI-based algorithms in a wide range of domains, including combinatorial optimization problems, function optimization, finding optimal routes, scheduling, structural optimization, image analysis, data mining, machine learning, bioinformatics, medical informatics, dynamical systems, industrial problems, operations research, and even finance and business.

The potential of SI is yet far from being exhausted with many interesting applications still to be explored, especially in bioinformatics. In the past few years, there has been a slow, yet steady increase in the number of research papers that have successfully applied SI algorithms in bioinformatics. This is because several tasks in bioinformatics involve optimization of different criteria (such as, energy, alignment score, overlap strength, etc.), and the various applications of SI algorithms proved them to be efficient, robust and computationally inexpensive optimization techniques, which made their applications in bioinformatics more obvious and appropriate. It is worth noting, however, that SI-based algorithms do not fully show their competitive edge over other optimization techniques on static problems whose characteristics and conditions do not change over time. Nevertheless, they are often more competitive to deterministic approaches in dealing with uncertainty, as well as general-purpose heuristics (e.g., hill climbing and simulated annealing approaches) in dealing with stochastic time-varying problem domains, due to their inherit adaptive capabilities.

General Advantages of Swarm Intelligence

a Scalability: SI systems are highly scalable; their impressive abilities are generally maintained when using groups ranging from just sufficiently few individuals up to millions of individuals. In other words, the control mechanisms used in SI systems are not too dependent on swarm size, as long as it is not too small.

b Adaptability: SI Systems respond well to rapidly changing environments, making use of their inherit auto-configuration and self-organization capabilities. This allows them to autonomously adapt their individual's behaviour to the external environment dynamically on the run-time, with substantial flexibility.

c Collective Robustness: SI Systems are robust as they collectively work without central control, and there is no single individual crucial for the swarm to continue to function (due to the redundancy of their individuals). In other words, the fault-tolerance capability of SI systems is remarkably high, since these systems have no single point of failure. A single point of failure is a part of any system that puts the entire system into risk of a complete failure, if it ceased to function.

d Individual Simplicity: SI systems consist of a number of simple individuals with fairly limited capabilities on their own, yet the simple behavioural rules at the individual level are practically sufficient to cooperatively emerge sophisticated group behaviour.

Software-Oriented Architecture: An Overview

Service-oriented architecture (SOA) is a software development model that allows services to communicate across different platforms and languages to form applications. In SOA, a service is a self-contained unit of software designed to complete a specific task. Service-oriented architecture allows various services to communicate using a loose coupling system to either pass data or coordinate an activity.

Coupling refers to the degree of direct knowledge that one element has of another while loose coupling is an approach to interconnecting the components in a system or network so that those components, also called elements, depend on each other to the least extent practicable. It also refers to a client of a service remaining independent of the service it requires. In addition, the client which can also be a service can communicate with other services, even when they are not related.

The Emergence of SOA

For decades, software development required the use of modular functional elements that perform a specific job in multiple places within an application. As application integration and component-sharing operations became linked to pools of hosting resources and distributed databases, enterprises needed a way to adapt their procedure-based development model to the use of remote, distributed components. Service-oriented architecture (SOA) emerged in the early part of the 21st century as an evolution of distributed computing [9]. Before SOA, services were understood as the end result of the application development process. In SOA, the application itself is composed of services. Services can be delivered individually or combined as components in a larger, composite service. Coming to interchanging SOA and web services [9] agreed that

SOA and web services are frequently conflicted, but they are not the same thing. SOA is an architecture that allows developers to combine multiple application services into a larger, composite service.

The general concept that emerged was the remote procedure call (RPC) which allowed one software process to invoke another as though it were local, even though the other process might be resident in a different application, computer or data centre. The problem with the general RPC model was that it allowed too much variation in implementation, and lacked:

a any systematic set of tools to provide security for the information flows
b governance with respect to access rights to the business processes involved or
c discovery, connection with, and message formats for, the services themselves.

While the concept of services introduced by SOA is an integral part of modern cloud computing and virtualization, SOA itself has been largely displaced.

Defining Concepts of SOA

a The business value is more important than the technical strategy.
b The strategic goals are more important than benefits related to specific projects.
c Basic interoperability is more important than custom integration.
d Shared services are more important than implementations with a specific purpose.
e Continued improvement is more important than immediate perfection.

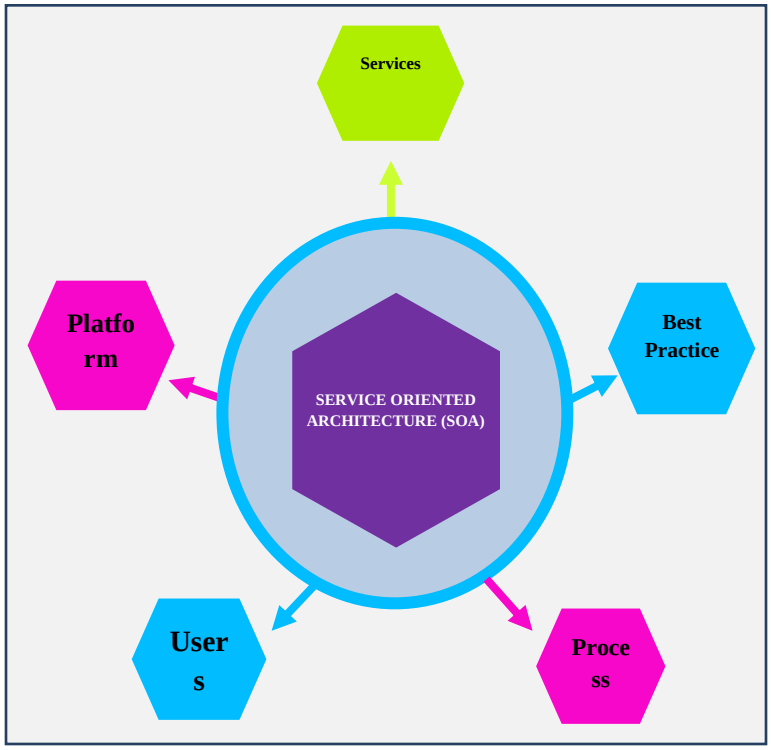

Fig.1. Generic Diagram of SOA (Google Image)

Service-oriented architecture is therefore an implementation of the "service concept" or "service model" of computing. In this style, business processes are implemented as software services, accessed through a set of strictly defined application program interfaces (APIs) and bound into applications through dynamic service orchestration. Furthermore, Service-Oriented Architecture (SOA) is an architectural approach in which applications make use of services available in the network. In this architecture, services are provided to form applications, through a communication call over the internet. SOA allows users to combine a large number of facilities from existing services to form applications. It encompasses a set of design principles that structure system development and provide means for integrating components into a coherent and decentralized system. SOA based computing packages functionalities into a set of interoperable services, which can be integrated into different software systems belonging to separate business domains.

Major roles within Service-Oriented Architecture

The [10] highlighted the duties of service provider and the consumer in the following points:

1. **Service provider:** The service provider is the maintainer of the service and the organization that makes available one or more services for others to use. To advertise services, the provider can publish them in a registry, together with a service contract that specifies the nature of the service, how to use it, the requirements for the service, and the fees charged.
2. **Service consumer:** The service consumer can locate the service metadata in the registry and develop the required client components to bind and use the service.

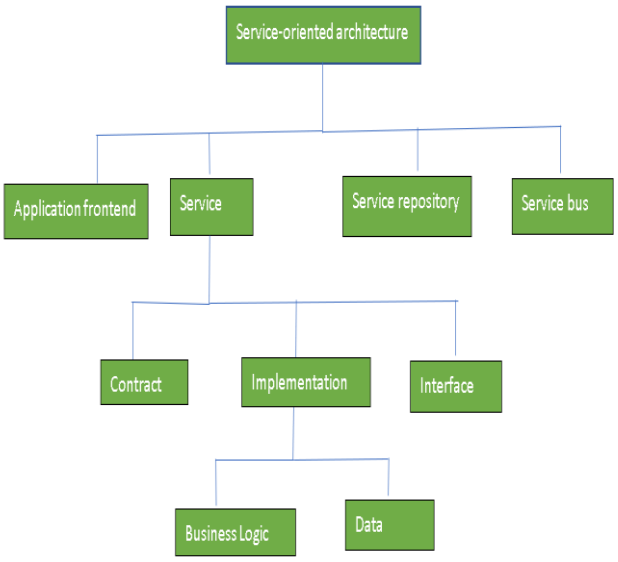

Fig. 2: Components of SOA (Source: [10])

Guiding Principles of SOA

According to [10] a highlight of the principle that should guide SOA was made and they include the following:

a Standardized service contract: Specified through one or more service description documents.
b Loose coupling: Services are designed as self-contained components and maintain relationships that minimize dependencies on other services.
c Abstraction: A service is completely defined by service contracts and description documents. They hide their logic, which is encapsulated within their implementation.
d Reusability: Designed as components, services can be reused more effectively, thus reducing development time and the associated costs.
e Autonomy: Services have control over the logic they encapsulate and, from a service consumer point of view; there is no need to know about their implementation.
f Discoverability: Services are defined by description documents that constitute supplemental metadata through which they can be effectively discovered. Service discovery provides an effective means for utilizing third-party resources.
g Composability: Using services as building blocks, sophisticated and complex operations can be implemented. Service orchestration and choreography provide a solid support for composing services and achieving business goals.

Practical Applications of SOA

SOA is used in many ways around us whether it is mentioned or not.

1. SOA infrastructure is used by many armies and air force to deploy situational awareness systems.
2. SOA is used to improve the healthcare delivery.
3. Nowadays many apps are games and they use inbuilt functions to run. For example, an app might need GPS so it uses inbuilt GPS functions of the device. This is SOA in mobile solutions.
4. SOA helps maintain museums a virtualized storage pool for their information and content.
5. SOA is useful in logistic firms whereby various apps represent different department which interact independently in carrying out logistic activities.

Advantages of SOA

a Service reusability: In SOA, applications are made from existing services. Thus, services can be reused to make many applications.
b Easy maintenance: As services are independent of each other they can be updated and modified easily without affecting other services.
c Platform independent: SOA allows making a complex application by combining services picked from different sources, independent of the platform.
d Availability: SOA facilities are easily available to anyone on request.

e Reliability: SOA applications are more reliable because it is easy to debug small services rather than huge codes
f Scalability: Services can run on different servers within an environment, this increases scalability

Web Services with Software Oriented Architecture

Web Services are self-contained, modular applications that can be described, published, located, and invoked over a network, generally, the World Wide Web. As earlier asserted, the Web Services architecture describes three roles: service provider, service requester and service broker; and three basic operations: publish, find and bind. Two separate documents describe Web Services: A Well-Defined Service (WDS) document describes non-operational service information, such as service category, service description, and expiration date, as well as business information about the service provider, such as company name, address, and contact information. A Network-Accessible Service Specification Language (NASSL) document describes operational information about the service, such as service interface, implementation details, access protocol, and contact endpoints.

A Web Services architecture implementation should allow for incremental security and quality of service models facilitated by configuring a set of environmental prerequisites (for example, authentication mechanism, billing, and so on) to control and manages the interactions. Web Services can be dynamically composed into applications stemming from capabilities-based look-up at runtime, instead of the traditional static binding. The dynamic nature of the collaborations allow the implementations to be platform- and programming language-neutral, and communications mechanism-independent, while creating innovative products, processes, and value chains. The service provider sends a WSDL file to UDDI. The service requester contacts UDDI to find out who is the provider for the data it needs, and then it contacts the service provider using the SOAP protocol. The service provider validates the service request and sends structured data in an XML file, using the SOAP protocol. This XML file would be validated again by the service requester using an XSD file.

Web Architecture Basics

In web architecture, we have the front-end and the back-end. The front-end hits the back end application while the back end hits the database and does some business logic. Then that back end returns the response to front end to render.

Fig. 3: Web Service Architecture (Source: Google Image)

The back-end is any application that is connected to the internet in the cloud whose primary purpose is service request e.g. web servers and databases. It is any application or set of applications connected to the internet whose primary purpose is service client request. Web server is anything that responds to http request with some response and return file to it. It is as simple as loading static file from a computer, return a file from computer. Database is a second back end process and can be used independently from web server. Here we think of backend as a huge system where many computers are communicating with each other. In Amazon for instance, you click on one button and about fifty processes are activated. The Front End can be anything running on a client, code running on your browser or on your phone or client and server mental model. A client can be the chrome browser on your phone. Client server relationship is the basic premise of the internet. You are the one pressing button on the front-end application.

The Web Service (WS)

The term Web Service (WS) is either a service offered by an electronic device to another electronic device, communicating with each other via the World Wide Web, or a server running on a computer device, listening for requests at a particular port over a network, serving web documents (HTML, JSON, XML, images), and creating web applications services, which serve in solving specific domain problems over the Web (WWW, Internet, HTTP). The term "Web service" describes a standardized way of integrating Web-based applications using the XML (Extensible Markup Language), SOAP(Simple Object Access Protocol), WSDL(Web Services Description Language) and UDDI(Universal Description, Discovery, and Integration) open standards over an Internet Protocol backbone. XML is the data format used to contain the data and provide metadata around it, SOAP is used to transfer the data, WSDL is used for describing the services available and UDDI lists what services are available. Furthermore, a Web service

is a method of communication between two electronic devices over a network. It is a software function provided at a network address over the Web with the service always-on as in the concept of utility computing.

All of these rules for communication are defined in a file called WSDL (Web Services Description Language), which has a .WSDL extension. (Proposals for Autonomous Web Services (AWS) seek to develop more flexible Web services that do not rely on strict rules.). A directory called UDDI (Universal Description, Discovery, and Integration) defines which software system should be contacted for which type of data. So, when one software system needs one particular report/data, it would go to the UDDI and find out which other systems it can contact for receiving that data. Once the software system finds out which other systems it should contact, it would then contact that system using a special protocol called SOAP (Simple Object Access Protocol). The service provider system would first validate the data request by referring to the WSDL file, and then process the request and send the data under the SOAP protocol.

Web Services components

Several essential activities need to happen in any service-oriented environment. A Web service needs to be created, and its interfaces and invocation methods must be defined.

i. A Web service needs to be published to one or more intranet or Internet repositories for potential users to locate.
ii. A Web service needs to be located to be invoked by potential users.
iii. A Web service needs to be invoked to be of any benefit.
iv. A Web service may need to be unpublished when it is no longer available or needed.

Advantages of Web Service

Use of the Web Services architecture provides the following benefits:

a Promotes interoperability by minimizing the requirements for shared understanding XML-based interface definition language (NASSL), an XML-based service description (WDS) and a protocol of collaboration and negotiation are the only requirements for shared understanding between a service provider and a service requester.

b Enables just-in-time integration. Collaborations in Web Services are bound dynamically at runtime. A service requester describes the capabilities of the service required and uses the service broker infrastructure to find an appropriate service. Once a service with the required capabilities is found, the information from the service's NASSL document is used to bind to it.Dynamic service discovery and invocation (publish, find, bind) and message-oriented collaboration yield applications with looser coupling, enabling just-in-time integration of new applications and services. This in turn yields systems that are self-configuring, adaptive and robust with fewer single points of failure.

c Reduces complexity by encapsulation. All components in Web Services are services. What is important is the type of behaviour a service provides, not how it is implemented. A WDS document is the mechanism to describe the behaviour encapsulated by a service.

d Encapsulation is key to:
   a. Coping with complexity. System complexity is reduced when application designers do not have to worry about implementation details of the services they are invoking.
   b. Flexibility and scalability. Substitution of different implementation of the same type of service, or multiple equivalent services, is possible at runtime.
   c. Extensibility. Behaviour is encapsulated and extended by providing new services with similar service descriptions.

e Enables interoperability of legacy applications. By allowing legacy applications to be wrapped in NASSL and WDS documents, and exposed as services, the Web Services architecture easily enables new interoperability between these applications. In addition, security, middleware and communications technologies can be wrapped to participate in a Web service as environmental prerequisites. Directory technologies, such as LDAP, can be wrapped to act as a service broker.

Web Application Architecture.

Web application architecture defines the interactions between applications, middleware systems and databases to ensure multiple applications can work together. When a user types in a URL and taps "Go," the browser will find the Internet-facing computer the website lives on and requests that particular page. The server then responds by sending files over to the browser. After that action, the browser executes those files to show the requested page to the user. Now, the user gets to interact with the website. Of course, all of these actions are executed within a matter of seconds. Otherwise, users wouldn't bother with websites. What's important here is the code, which has been parsed by the browser. This very code may or may not have specific instructions telling the browser how to react to a wide swath of inputs. As a result, web application architecture includes all sub-components and external applications interchanges for an entire software application.

Of course, it is designed to function efficiently while meeting its specific needs and goals. Web application architecture is critical since the majority of global network traffic, and every single app and device uses web-based communication. It deals with scale, efficiency, robustness, and security. With web applications, you have the server vs. the client side. In essence, there are two programs running concurrently:

a The code which lives in the browser and responds to user input

b The code which lives on the server and responds to HTTP requests

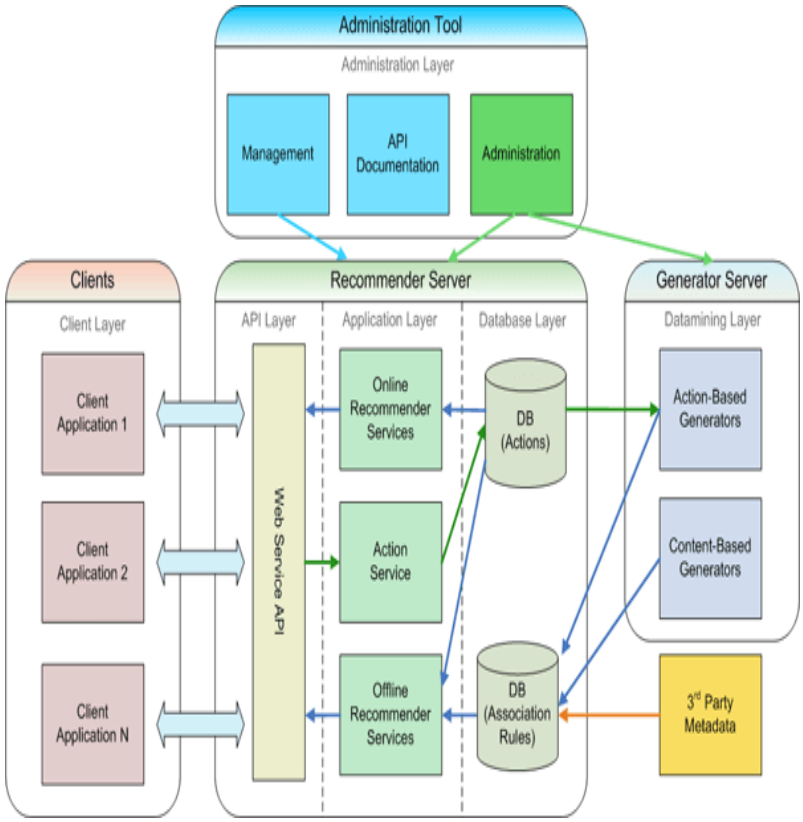

Fig. 4 Web Application Architecture (Source: [11])

Trends in Web Application Architecture

As technology continues to evolve, so does web application architecture. One such trend is the use of and creation of service-oriented architecture. This is where most of the code for the entire application exists as services. In addition, each has its own HTTP API. As a result, one facet of the code can make a request to another part of the code–which may be running on a different server. Another trend is a single-page application. This is where web UI is presented through a rich JavaScript application. It then stays in the user's browser over a variety of interactions. In terms of requests, it uses AJAX or WebSockets for performing asynchronous or synchronous requests to the web server without having to load the page.

The user then gets a more natural experience with limited page load interruptions. At their core, many web applications are built around objects. The objects are stored in tables via an SQL database. Each row in a table has a particular record. So, with relational databases, it is all about relations. You can call on records just by listing the row and column for a target data point. With the two above trends, web apps are now much better suited for viewing on multiple platforms and multiple devices. Even when most of the code for the apps remains the same, they can still be viewed

Related Work

In this section, we will review some related researches and applications developed which are related to our current study.

- Oliveira et al [12] proposed Information and Service-Oriented Architecture & Web Services: enabling integration and organizational agility. They discussed the key technologies and problems of processes models and organizational modeling construction. They also proposed new information architecture which will suit globalized markets changes (even in the maturation process), and constitute one of the essential requirements for online information architecture.
- Altalabani and Khaddaj [13] proposed a framework for the testability of service oriented architecture. They discussed the proposal of an approach which will make SOA testing on real-time, much relevant and applicable. However, they were unable to implement a fully automated testing approach for testing SOA system.
- Pham et al [14] proposed swarm intelligence for next generation wireless networks. Their primary focus was the integration of two domains: NGN and SI. First, they provided an overview of SI techniques from fundamental concepts to well-known optimizers. Secondly, they reviewed the applications of SI to settle emerging issues in NGN, including spectrum management and resource allocation, wireless caching and edge computing, network security, and several other miscellaneous issues. Finally, they highlighted open challenges and issues in the literature.
- Yamid et al [15] proposed a computational intelligence-based framework for the development of Business intelligence using service oriented architecture. The research method was an applied research. They presented the development of a framework for software development using computational intelligence. The results showed that the proposed intelligent framework for Business Intelligence development systems could be considered as a tool for making business decisions.
- Pahl [16] proposed a semantic model driven architecting of service-based software systems. In their study, they presented ontology-based transformation and reasoning techniques for layered semantic service architecture modeling. Their approach addressed a software service-specific solution, reflecting the current development of the Web services and the Semantic Web. The primary platform they aimed to support was service-oriented architecture with the second Web services generation focusing on processes and composition, utilizing the Semantic Web with its ontology technology support. However, they were unable to develop a model that will maximize the benefits of the web service architecture.
- Macro et al. [17] they carried out a systematic review of previous studies in the field of AI and also reviewed five interviews with software engineers who are in the field of AI to assess the status of development, future development potentials and equally the risks of AI application to software engineering.
- Lodha and Sharma [18] proposed a design and implementation of service-oriented architecture. They identified the full service approach that offer Integration. In this method, a process model of organization and inventory lists organizations as inputs received during a three-step process for integrating services organizations were recognized organization.

- Arch-int [19] proposed a graph-based semantic web service composition for healthcare data integration. The management time subsystem is responsible for dependency graph preparation in which a dependency graph of related services is generated automatically according to the proposed semantic matchmaking rules. The run-time subsystem is responsible for discovering the potential web services and non-redundant web services composition of a user's query using a graph-based searching algorithm. The proposed approach was applied to healthcare data integration in different health organizations and was evaluated according to two aspects: execution time measurement and correctness measurement.
- Aljebry and Qureshi [20] presented a proposal to improve mobility in service oriented architecture. They integrated two enhancement SOA architectures and added a new component to that architecture called SU as a solution for this restriction. It has many benefits such as give Mobile Web Service Verifier component action procedures in a situation of limited service in SP, raise the service creation speed and satisfy the consumer service needs.
- Lakta and Provost [21] proposed an implementation of a cloud-based service-oriented architecture for hardware control systems supported by neural network. They described a new approach of controlling mobile robots through a cloud-based service-oriented architecture supported by neural networks. The proposed architecture permits to operate with large and complex networks even on mobile robotic systems as illustrated on a case-study. Therefore the proposed architecture was based on dividing the management for the high-level on a computer or server and for the low-level directly on the mobile robot. Aiming at copying the cognitive capabilities of human beings, the system is reducing the amount of required sensors significantly. Successfully applied experimentally, the proposed architecture could also be replicated in various other industrial situations.
- Pablo et al [22] carried out a study on the developing services in a service-oriented architecture for evolutionary algorithms. They designed and implemented services for Evolutionary Computation in the Service Oriented Architecture paradigm. This paradigm allows independence in language and distribution, but the development requires to manage some technological and design issues, such as abstract design or unordered execution. To solve them, OS-GiLiath, an implementation of an abstract Service Oriented Architecture for Evolutionary Algorithms, is used to develop new interoperable services taking into account these restrictions.
- Nahvi and Habibi [23] proposed a study on improving Service-Oriented Architecture Processes in Process of Automatic Services Composition Using Memory and QF, QWV Factor. In the proposed framework, elements affecting management of service-oriented architecture processes were organized according to a logical procedure. This framework identified processes of this style of architecture based on requirements in service-oriented architecture processes management and according to qualitative features in this area. In the proposed framework, in addition to using existing data in the problem area, existing structure and patterns in the area of software architecture were also utilized, and management processes in service-orientated architecture are improved based on propriety of available requirements.

- Avila et al [24] proposed applications based on service-oriented architecture (SOA) in the field of home healthcare. A literature review of applications developed in the health industries which are focused on patient care from home and implement a service-oriented (SOA) design in architecture was carried out. Throughout the work, the applicability of the concept of Internet of Things (IoT) in the field of telemedicine and health care in general was evaluated.

Our Focus

We have reviewed some literatures related to our study of discuss. However, we are more interested in the work carried out by Nahvi and Habibi [23]. They proposed a study on improving Service-Oriented Architecture Processes in Process of Automatic Services Composition Using Memory and QF, QWV Factor. In the proposed framework, elements affecting management of service-oriented architecture processes were organized according to a logical procedure. This framework identified processes of this style of architecture based on requirements in service-oriented architecture processes management and according to qualitative features in this area. However, they could not implement an action procedure for scenarios where a required web service is unavailable and they could not also implement automatic update of services to adapt to new external environment. We hope to improve on the work carried out by Nahvi and Habibi [23] by solving these problems using swarm intelligence technique.

Methodology

The methodology adopted in this research is the Object-oriented Design Methodology (OODM). OODM is a process of planning a system of interacting objects for the purpose of solving a software problem. It is one approach to software design. An object contains encapsulated data and procedures grouped together to represent an entity. The object interface defines how the object can be interacted with. In relation to our research, the objects are the web services which are integrated together to solve a problem or meet a service request. This makes this methodology most suited for this research.

Analysis of the Existing System

The existing system was developed by Nahvi and Habibi [23]. The model was composed of components and depositories. Each component was a computing entity which was an interconnected collection of the system functionality requirements. Components communicated through one or more ports to the environment. Ten categories of agents helped components so that system requirements were removed. The overall structure of the existing framework is shown in Figure 5. Business Process Management (BPM) is taken from stakeholders by user interface agent and it is stored in this component under Request For Change (RFC) standard form. Requirements in this framework are stated under title of RFC so that it can be integrated to ITIL framework in the future. Problem definition in the framework was defined by RFC which is an ordered triple: RFC (T, QWV, C).

In the existing framework, each service was connected to a standardizer known as common rules database (CRD). This unit was a basis of existing rules and norms, related to application case, which was embedded in the proposed framework in order to chat and negotiate with CRD connected to different services. Existing framework was shaped based on reference layering of service-oriented architecture. Multi-agent system was used for improving service-oriented architecture processes. In evaluator component, registry agent scores services, business processes and solutions considering their qualitative features, and turns QoS value from static to dynamic state.

Disadvantages of the Existing System

Although the existing system has added new features to the web service architecture, yet it has the following limitations: The existing system does not provide an action procedure for tackling unavailable web services when they are requested.

a. The existing system does not automatically update services or components to adapt to new external environments when request and invoked.
b. The existing system does not include an intelligent technique that aids components to interact with each other locally to aid optimization.

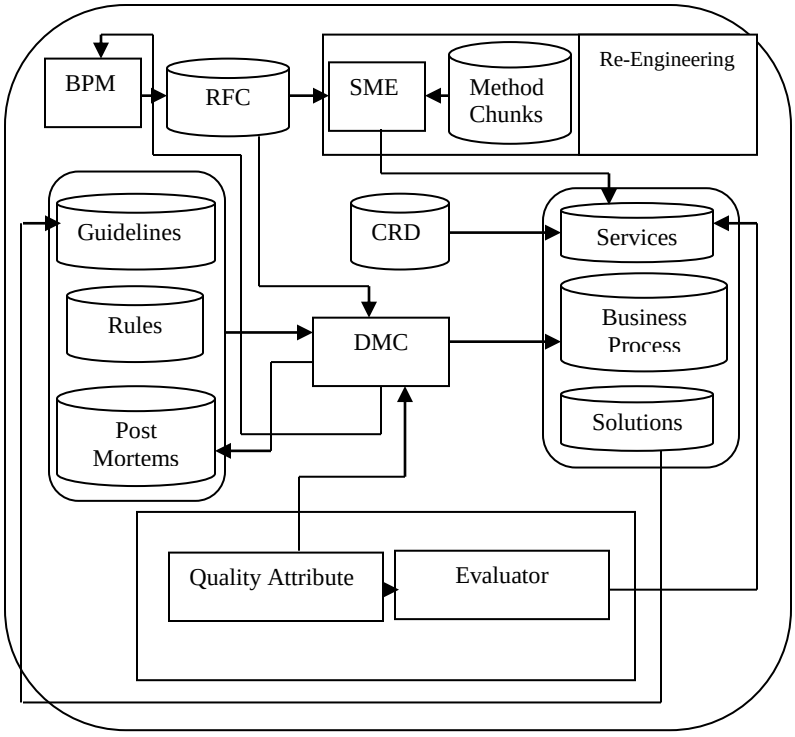

Fig. 5: Architecture of the Existing System (Source: [23])

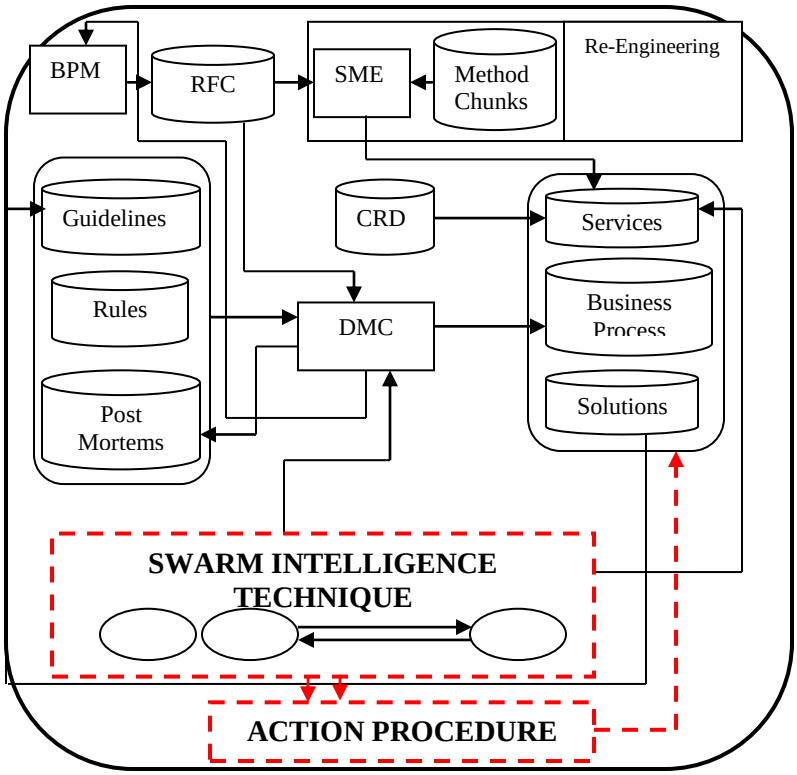

Fig. 6 Architecture of the Proposed System

Analysis of the Proposed System

The proposed system is an improvement of the work carried out by [23]. The main idea of the proposed system is the introduction of an intelligent technique for component interaction in the systems when integrated together. The proposed system also centres on the implementation of an action procedure that handles missing web service scenarios. The intelligent technique introduced in this study is the swarm intelligence technique. This technique draws it origin from biological creatures such as bees. The idea behind this technique is that it focuses on the interaction of a group of related objects (in this case bees) to achieve a common goal.

In typical web architecture, requests are issued and services are granted upon the request, but in a service oriented architecture, services or components are integrated together to form a new system instead of starting from the scratch to build that system. This means that services or components are reusable, and the same services or components form the fundamental framework of several applications. These services are transmitted over a network with a communication protocol. However, there are scenarios when a particular component that is required by another application is not available, the system will face a major drawback in this case as most service-oriented web architectures do not provide any action procedure to handle this situation. The proposed system provides a procedure that will be triggered in this case, this procedure will search for

other similar services to the requested service, and if no service is available, it will automatically modify a component to meet the need of the request.

Advantages of the Proposed System

a. The proposed system provides an action procedure for tackling unavailable web services when they are requested.
b. The proposed system automatically updates services or components to adapt to new external environments when request and invoked.
c. The proposed system includes an intelligent technique that aids components to interact with each other locally to aid optimization.

System Algorithm

This section illustrates the steps used for the system implementation using the PHP language.

Algorithm of the Existing system

Improving Service-Oriented Architecture Processes in Process of Automatic Services Composition Using Memory and QF, QWV Factor (Nahvi and Habibi [23])

Step 1: Declare Variables

RFC, Si, SDSi, ti, WSLi, SLi, FLi, QFRL, QWV, qw 1, qw 2, CSLi.

WHERE
RFC = User Request
Si = web service
SDSi: = sub domain of Si
ti = Task involved in user request
WSLi = list of web services in the registry table
SLi = search list
FLi = filtered list
QFRL = QF ranked List
QWV = QoS weighted vector
(QWV⟩ = ⟨qw1, qw 2⟩
qw 1 = Cost Weight
qw 2 = Response Time Weight
CSLi = composed Services List (Register in solution repository)
Begin

STEP 2: For each task ti in RFC
STEP 3: Discover (WSLi, SDSi)
STEP 4: For each Si in SLi
Do
STEP 5: If (Availability == true) & (QF>0)
STEP 6: SL.add()
STEP 7: Endif

STEP 8: EndFor
STEP 9: QoS based Service Selection (SL)
STEP 10: Compute QoSRank(FL)
STEP 11: Final Rank based Sorting (QFRL,QWV)
STEP 12: Apply Genetic Algorithms
STEP 13: End For
STEP 14: Return CSL
End

Algorithm of the Proposed System

An Improved Model for Service-Oriented Web-Architecture Using Swarm Intelligence Technique.

STEP 1: Declare Variables
RFC, Si, SDSi, ti, WSLi, SLi, FLi, QFRL, QWV, qw 1, qw 2, CSLi, AP, Sit,QoS
WHERE
RFC = User Request
Si = web service
SDSi: = sub domain of Si
ti = Task involved in user request
WSLi = list of web services in the registry table
SLi = search list
FLi = filtered list
QFRL = QF ranked List
QWV = QoS weighted vector
(QWV⟩ = ⟨qw1, qw 2⟩
qw 1 = Cost Weight
qw 2 = Response Time Weight
CSLi = composed Services List (Register in solution repository)
AP = Action Procedure
Sit = Swarm Intelligence technique
QoS = Quality of Service
Begin
STEP 2: For each task ti in RFC
STEP 3: Discover (WSLi, SDSi)
STEP 4: For each Si in SLi
Do
STEP 5: If (Availability == true) & (QF>0)
STEP 6: SL.add()
STEP 7: Launch AP
STEP 8: AP = search SLi for related services or components

STEP 9: IF search = 0;
STEP 10: modify Si to match request.
ELSE
STEP 11: IF search = 1;
STEP 12: execute service request
STEP 13: QoS based Service Selection (SL)
STEP 14: Compute QoSRank(FL)
STE 15: Final Rank based Sorting (QFRL,QWV)
STEP 16: Apply SIt
STEP 17: SIt = auto-update Si to match new environment
STEP 18: Maximize QoS
End

Discussion of Results

The Figures illustrated below are the program output from the implementation of the proposed system. The system was implemented with web based languages such as PHP and JavaScript. MySQL database was used to store the protocols for communications between the system and other applications during service request. Xampp Local Host server was used to host the system locally during testing. The implemented system is a simulation of the original software and will require certain upgrade features before hosting it on the main web i.e. the Internet.

Fig. 7: Welcome Page

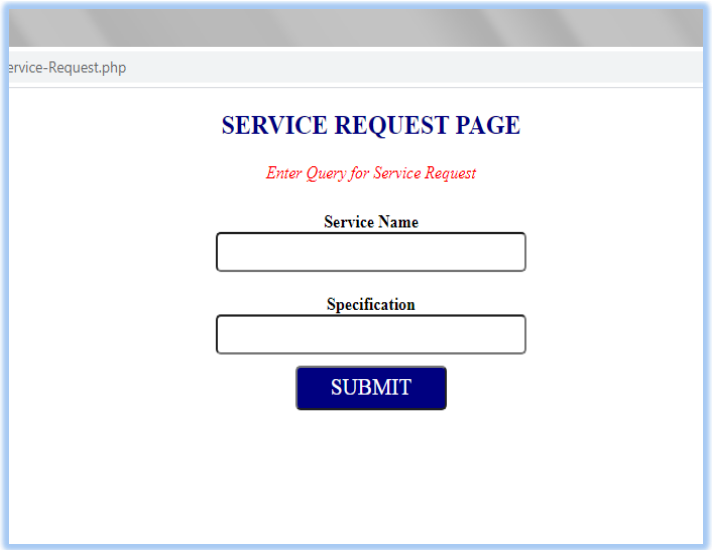

Fig 8: Service Request Page of the Proposed System

The service request page is a platform that allows users to search for particular services that they need to carry out their service request or integrate to the application under development. The search is carried out by specifying the name of the service and the specification. The specification is the extension of the service such as *.xml* etc. Once this request is issued, the system searches the service repository for all services with that name and specification, and displays all the possible matches. If no match is found for the inputted search, the alert in Fig 9 is displayed. The system further prompts the user to trigger the action procedure as shown in Fig 10.

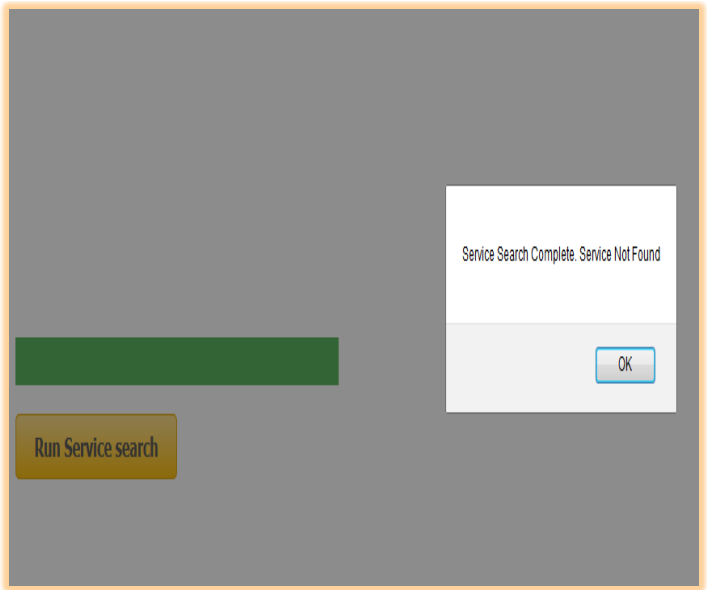

Fig 9: Service Request Fail Page: Service not Found Prompt

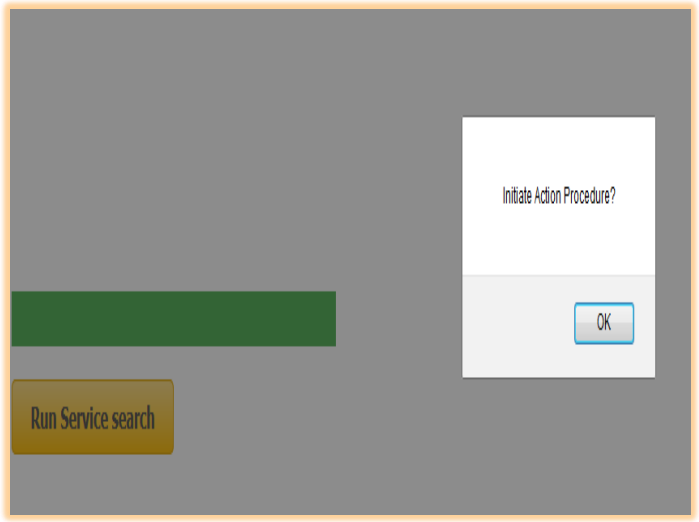

Fig 10: Action Procedure Launch Prompt

If this prompt is clicked, the system searches for closely related matches, and if none is found, it modifies a service to match the requirement of the request and submits the service to the requesting application.

Fig 11: Service Request Success Page.

When the results of the system testing was evaluated based on certain parameters such as speed of computation, model efficiency, cross platform adaptability, the proposed system was found to be more efficient than the existing, with an efficiency score of 86% against 66% of the existing system.

Table 1: Comparative Analysis

SN	EXISTING SYSTEM	Score	Score	PROPOSED SYSTEM
1.	Speed in Processing inputted Search Request	20 (seconds)	12 (seconds)	Speed in Processing inputted Search Request
2.	Speed in Search	14 (seconds)	8 (seconds)	Speed in Search
3.	Cross Platform Compatibility (CPA)	11	20	Cross Platform Compatibility (CPA)
4	Model Efficiency (ME)	10	23	Model Efficiency (ME)
5	Cost Benefit Analysis (CBA)	11	23	Cost Benefit Analysis (CBA)
Total		66%	86%	**Total**

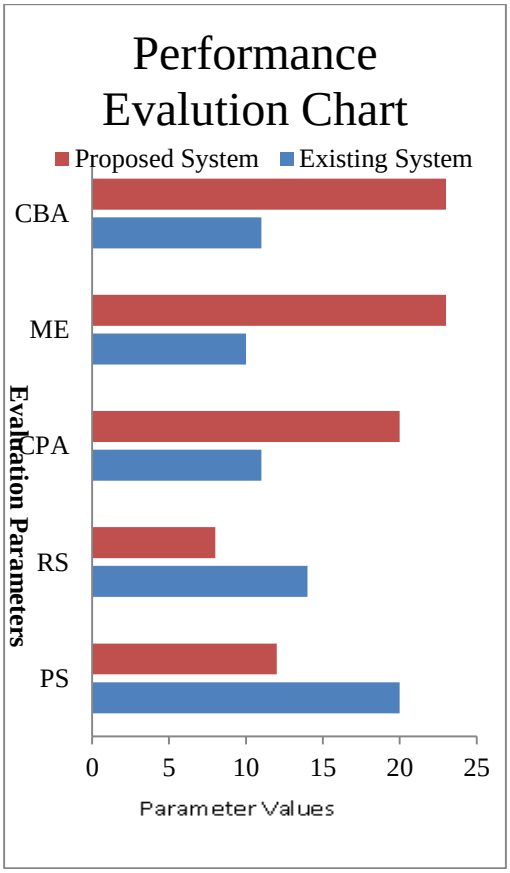

Fig. 12: Performance Evaluation Chart

Conclusion

Swarm intelligence technique is an emerging aspect of artificial intelligence that enables inter-relationships between components in the same local environment to foster optimization and yield higher productivity. This technique was adopted in the improvement of the benefits of service-oriented web architecture. The technique was used to build an improved model for service-oriented models. Two main features were added to the service oriented architecture in this study. The first is the action procedure that handles service request which are unavailable. Secondly, an automatic update of the services to adapt to new environments in other applications upon invocation. The results show that the time used for meeting service request and the cost of building new applications are relatively cheaper using this model. Our future work will be based on using a hybrid model based on artificial intelligence such as Swarm Intelligence and Ant bee Colony techniques to enhance the service-oriented web architecture.

References

[1] A. Abraham, H. Guo, &H. Liu. "Swarm Intelligence: Foundations, Perspectives and Applications." DOI: 10.1007/978-3-540-33869-7_1. 2006.

[2] H. Ahmed, and J. Glasgow, "Swarm Intelligence: Concepts, Models and Applications." *Technical Report.* Vol. 585. (2012).

[3] Y. Zhang, P. Agarwal, V. Bhatnagar, S. Balochian. &J. Yan,. "Swarm intelligence and its applications." *The Scientific World Journal*. Vol. 1, Issue 3. 2013. http://dx.doi.org/10.1155/2013/528069.

[4] S. Cheng, S. "Special issue on swarm intelligence in big data analytics." *International Journal of Swarm Intelligence Research (IJSIR).*2013. https://www.researchgate.net/publication/251236723_International_Journal_of_Swarm_Intelligence_Research_IJSIR_Special_Issue_on_Swarm_Intelligence_in_Big_Data_Analytics/link/0deec51ef7ed3edd6d000000/download

[5] L. Rosenberg. "Artificial swarm intelligence." 2019. https://11s1ty2quyfy2qbmao3bwxzc-wpengine.netdna-ssl.com/wp-content/uploads/2019/01/Artificial-Swarm-Intelligence-white-paper-1-21-19.pdf/

[6] European Commission On artificial intelligence. "A European approach to excellence and trust." *White Paper*. 2020. https://ec.europa.eu/info/sites/info/files/commission-white-paper-artificial-intelligence-feb2020_en.pdf.

[7] M. Purdy, R. Eitel-Porter, and M. Klymenko. How swarm intelligence blends global and local insight. 2020. https://sloanreview.mit.edu/article/how-swarm-intelligence-blends-global-and-local-insight/?gclid=Cj0KCQiAy579BRCPARIsAB6QoIbK9tXedMUQiooMWE5BU1bCUHTtSJVrqRrADIwYbRs1qq1cl3j_5HAaAqwcEALw_wcB

[8] A. Biswas, and B. Biskar. "Swarm Intelligence Techniques and Their Adaptive Nature with Applications." 2015https://doi.org/10.1007/978-3-319-12883-2_9.

[9] M. Tyson. What is service-oriented architecture? 2020.

[10] Geeks for Geeks, "Service-oriented architecture." (2018). https://www.geeksforgeeks.org/service-oriented-architecture/.

[11] Stackify. "What is web application architecture?" https://images.app.goo.gl/oDU7VijTGWhUpgHw9 2020.

[12] S.B. Oliveira, A.J. Balloni, F.N. Barbara and F.A. Toda. "Information and Service Oriented Architecture and Web Services: Enabling Integration and Organizational Agility." *Elsevier.* Vol. 5. Pp. 141-151. 2012.

[13] O. Altalabani and S. Khaddaj. "A Framework for the Testability of Service-oriented Architecture." *Journal of Algorithm & Computational Technology*. Vol. 3, No. 3. Pp.455-471. 2012.

[14] Q.V. Pham, C.D. Nguyen and S. Mirjalili. Swarm Intelligence for Next Generation Wireless Networks: Recent Advances and Applications. *Research-gate Publication.* pp. 1-30. 2020.

[15] F.H. Yamid, M.A.J. Paba, N.E.L. Narvaez, H.M Hernandez, and W.N. Bernal. Framework for the Development of Business Intelligence using Computational intelligence and Service-oriented Architecture. pp. 1-7.

[16] C. Pahl. Semantic Driven Architecting of Service-Based software systems. pp. 1-23.

[17] B. Macro, J. Rebstadt and O. Thomas. "Application of AI in Classical Software Engineering." *All Aspects.* Vol. 2, No.1. pp. 1-16. 2020.

[18] G.D. lodha and A. Sharma. "Design and Implementation of service-oriented Architecture." *International Journal of Advanced Research in science and Engineering (IJARSE).* Vol. 5, No.4. 2016.

[19] N. Arch-int, S. Arch-int, S. Sonsilphong and Wanchai. 2017. Graph-Based Semantic Web Service Composition for Healthcare Data Integration. *Hindawi Journal of Healthcare Engineering.* Pp. 1-19. 2017.

[20] D.A. Aljebry and R. Qureshi. "Proposal to Improve Mobility in Service Oriented Architecture." *International Journal of Education and Management engineering.* Vol. 1. Pp. 14-24. 2017.

[21] O. Latka, and J. Provost. "Implementation of a Cloud-based Service-Oriented Architecture for Hardware Control Systems supported by Neural Network". Pp. 1-8. 2017.

[22] G.S. Pablo, M. I. García Arenas, A.N. Mora, P.A. Castillo, C. Fernandes, P. Cuevas, J. González and J.J. Merelo. Developing Services in a Service Oriented Architecture for Evolutionary Algorithms. Pp. 1-8. 2013

[23] B. Nahvi and J. Habibi. "Improving Service-Oriented Architecture Processes in Process of Automatic Services Composition Using Memory and QF, QWV Factor". *International Journal of Advanced Computer Science and Applications (IJACSA).* Vol. 7, No. 6, pp. 189-197. 2016

[24] K. Avila, P. Sanmartin, D. Jabba and M. Jimeno. "Applications Based on Service-Oriented Architecture (SOA) in the Field of Home Healthcare."*Sensors.* Vol. 17, No. 1703, pp. 1-16. 2017.