International Journal of Current Engineering and Technology
E-ISSN 2277 – 4106, P-ISSN 2347 - 5161

Available at http://inpressco.com/category/ijcet

Research Article

Enterprise Application Integration using Service Oriented Architecture with Generic Pattern

Nilesh Vishwasrao Patil[Å*], M. C. Kshirsagar[Å] and P. C. Jaypal[Å]

[Å]Vishwbharati Academy College of Engineering, Ahmednagar, Pune University

Accepted 01 Oct 2014, Available online 10 Oct 2014, **Vol.4, No.5 (Oct 2014)**

Abstract

Nowadays the computer world is almost migrating from the tightly coupled architecture to loosely coupled architecture directly or indirectly. There are many intentions of Enterprise Application Integration in the organizations to achieved desired functions. The main objective of the Enterprise Application Integration is to implement integration layer between heterogeneous and/or homogeneous systems using Service Oriented Architecture in the direction of achieving the loosely coupled architecture. The integration architecture we have today is very limited and need to improve it as agile as possible. The reusability is not the new thing for software industry; it has been playing the crucial part in the software life cycle development process from last decade. Software industry has achieved the powerful accomplishment in throughput and preeminence because of reusability in software development process instead of creating same thing again and again. In this paper, we will discuss about Enterprise Application Integration using Service Oriented Architecture for achieving loosely coupled architecture directly or indirectly with the help of Web Services or Messaging Queue services.

Keywords: *SOA: Service Oriented Architecture, WS: Web services, WSDL: Web Service Description/Definition Language, W-SOA: Web Service Oriented Architecture, ESB: Enterprise Service Bus.*

1. Introduction

Now a days the IT industries is experiencing from major changes because the development of new technologies, tools, moving data towards the big data (unstructured data) etc. The software industries are almost migrating from the tightly coupled architecture to loosely coupled architecture directly or indirectly. Present software systems must follow to requirements such as flexibility, adaptability, time to market, and continuous business process reengineering. Determined by these significant requirements for modern system, we need to migrate of legacy systems towards the new network-centric operating platforms and their consequent integration with other back-end applications using web technologies is become a very effective strategy for many organizations to maintain a competitive edge. This approach emphasis on leveraging existing legacy software assets by minimizing the risks involved in re-implementing large-scale mission-critical legacy applications. The most important thing is organization could not want to replace the existing systems from their business because of huge amount of money had invested and large amount of data had stored but they also want to grab advantages of new systems. Then we will need to implement integration layer between them, that we will discuss in latter in this paper. That means we will not going to replace legacy system, just to build bridge between different applications (Legacy system, RDBMS, No SQL databases etc.)

The merging of the Internet and distributed-object technologies extends this information based internet to a worldwide services-based web, where software services and content are distributed openly over the internet, intranets, and extranets. Such distributed services are specified in the form of components in the web domain and can be dynamically deployed by a web-server upon the request of arbitrary web clients. In this context, non-functional requirements such as interoperability, flexibility, customizability, and conformance with continuous business process reengineering activities, are fundamental for the successful deployment and integration of services in a web-based environment. As an example, consider a scenario in the new services based web, whereby a global infrastructure enables software components that have been developed independently; be integrated with each other in order to facilitate complex business tasks. In this way, data and, software components located within an organization, can be combined on an as required basis, forming thus collaborative information systems. However, systems interactions by using standard protocols. The full potential of web services as an integration platform will be achieved only when applications and business processes are able to integrate their complex interactions by using a standard process integration model.

*Corresponding author **Nilesh Vishwasrao Patil** is a ME Student; **M. C. Kshirsagar** and **P. C. Jaypal** are working as Assistant Professor

All electronic business is hungers to develop the systems or software within time by providing the cost benefit and efficiency to the client as well our self. To achieve such reusable functionality we have to identified the general things and create general patterns for regular occurring problems.

Reusability is the key to growth in any area including or excluding software field. Reusability means previously developed ideas, systems, software, tools and products reuse for avoid to implementing everything from scratch. Development of software, reuse the knowledge in the form of experience, processes in the form of methods, and product in the form of tools. To achieve systematic way of reusing the existing products, systems, processes and knowledge in electronic business to get maximum cost benefit, time benefit, standard and efficient template.

2. Background

A Service is independent, self-contained unit of functionality that will have organized, deployed, published and invoked over the set-up, cluster, web, grid, network etc. with loosely coupled infrastructure. SOA is utilizes networked services as important elements to build and integrate homogeneous and heterogeneous enterprise applications with minimizing most important economic factor and achieve loosely coupled architecture. All components are loosely coupled means each is independent to one another. As market is growing, big enterprises are used around thousands of applications. These applications can be new technology based applications or Legacy systems applications. Legacy systems form the backbone of the information flow within a big enterprise and these are central entity for associating information about corporate. We can then integrate these enterprise applications using Service Oriented Architecture (SOA) with the help of web service and MQ (Messaging Queue) based interfaces by building bridge between these applications for interaction.

Following are some important points for enterprise application integration using service oriented architecture:

Service Oriented Architecture

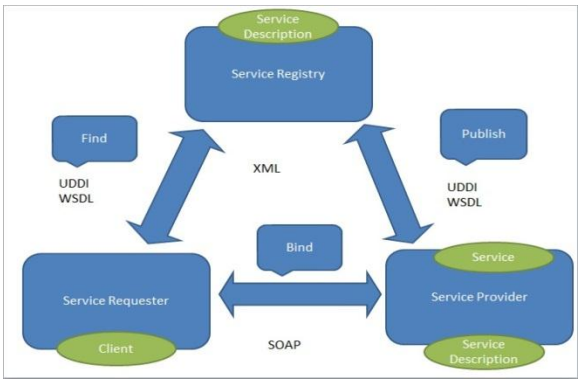

Fig. 1 Service Oriented Architecture

Service Oriented Architecture is collections of services that are able to communicate with each other. In this everything has been process through the services. A Service is independent, self-contained unit of functionality that will has organized, deployed, published and invoked over the set-up, cluster, web, grid, network etc. once services has been deployed then we can say to those services as interfaces. These interfaces can be implemented as web service or messaging queue series based to integrate systems disparate systems for achieving loosely coupled architecture. SOA is an approach through which integrate enterprise applications by achieving loosely-coupled interfaces/services.

It is considered to be the new phase in the evolution of distributed enterprise applications. In our more and more modest commercial world, in which business needs and enterprises are shifting continually, the quick progress of business applications is a key dynamic in the achievement of a business. SOA has promised to provide enterprises with flexible and extensible architectures that would enable them to adapt their applications easily so that they remain competitive and compliant. There are many of the major software sellers sell Service Oriented Architecture products such as Enterprise Service Buses (ESBs). Examples: IBM Web Sphere Message Broker, Oracle Fusion, Software AG Web Methods, Mule ESB etc.

Even though there is a common acceptance of this concept, a real problem hinders the widespread use of SOA: A methodology to design and build secure SOA-based applications is still needed. Communications among services can involve only simple invocations and data passing or complex coordinated activities of two or more services. It has also provides characteristics like Abstraction, Encapsulation, Reusability, Loosely coupled architecture, integrate disparate systems, Discoverability etc. The main implementation platform for Service Oriented Architecture is web services and messaging queue based.

Web Services

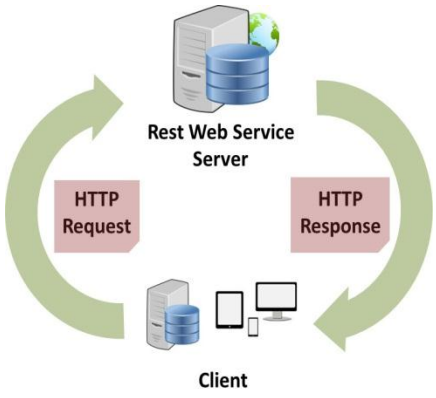

Fig. 2 Web Services

A Web service is a process 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. Web service is a kind of service which is language independent so that it is interoperable between different communications protocol, diff. operating systems, heterogeneous systems and various programming

languages. Now a days web services has been playing the important role in current electronic business Web service is also plays the important role in Enterprise Application Integration.

```
<?xml version="1.0"?>
<definitions name="StockQuote"

targetNamespace="http://example.com/stockquote/service"
          xmlns:tns="http://example.com/stockquote/service"
          xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
          xmlns:defs="http://example.com/stockquote/definitions"
          xmlns="http://schemas.xmlsoap.org/wsdl/">

  <import namespace="http://example.com/stockquote/definitions"
          location="http://example.com/stockquote/stockquote.wsdl"/>

    <binding name="StockQuoteSoapBinding" type="defs:StockQuotePortType">
        <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
        <operation name="GetLastTradePrice">
           <soap:operation soapAction="http://example.com/GetLastTradePrice"/>
           <input>
               <soap:body use="literal"/>
           </input>
           <output>
               <soap:body use="literal"/>
           </output>
        </operation>
    </binding>

    <service name="StockQuoteService">
        <documentation>My first service</documentation>
        <port name="StockQuotePort" binding="tns:StockQuoteBinding">
           <soap:address location="http://example.com/stockquote"/>
        </port>
    </service>
</definitions>
```

Fig. 3 Sample WSDL

For implementing the web service we will require the web service description/definition language and bring up as WSDL. The model of the WSDL is as shown in above fig 3.

Web service is one of the sorts of service which has following features:

XML based

Web Services has been used XML language at data representation and data transportation layers. XML is language and platform independent language so we don't care about operating system, platform binding. So that web services based applications is highly interoperable application.

Loosely Couple

In tightly coupled architecture, the client, business and server logic are closely tied with one another, indicating that if one interface is changed, that will effect on to the other interface. In Loosely coupled system even changed multiple interfaces which does not effect to the other interfaces.

Automatic Discovery

The service provider publishes the service description (Web Service Definition Language) in a service registry (UDDI), using the standard UDDI specification (Universal Description Discovery Integration). The requester systems (client) search the registry which needed for requester as per service specification. In registry, many service providers could be exists for a given service specification. Dynamically the client binds to a particular service provider.

Self-Describing

The functional specification of the service is expressed with of a XML-based descriptor. WSDL (Web Service Description Language) is the standard used for describing services.

Platform-agnostic

Services can be deployed on heterogeneous platforms and operating systems. Even the client and server platforms can be heterogeneous. The client and server exchange XML messages over SOAP-HTTP. Adoption of open XML standards is the primary difference between Web Services and other remote object technologies like RMI, COM etc.

Separation of interface from implementation

The service interface is disparate from the service implementation. Many providers can be exist for the given service interface.

Ability to synchronize and asynchronies

Web service has ability both synchronous and asynchronous way of invocation. Synchronous nature means only one request is process at time, other requester wait until previous request finishes. Asynchronous nature means multiple requesters can invoke the same service at time.

Support Remote Procedure call

Web services have allowed clients to invoke procedures which on remote objects with the help of XML protocol. A web service is supports Remote Procedure Call by providing services of its own.

Network, operating system and platform independent

Web services are network, language, platform independent because it based on XML so it is interoperable between heterogeneous systems.

Open standard based

XML and HTTP are the major technical foundation for Web services. It has been built using open-source projects.

Dynamic

Dynamic e-business can become reality using Web services because with UDDI and WSDL you can automate the Web service description and discovery.

Service Oriented Architecture (SOA) is a model in which Enterprise Applications Integration are constructed by

loosely-coupled Web-Services and MQ Series. An important characteristic of SOA model is automatic discovery and binding of Services is done.

Three important components of the Service Oriented Architecture are shown in fig 1: Service Provider, Service Requestor and Service Registry/Broker.

1) Service Provider

The service provider builds a Web service which publishes to the service registry. Provider has provided WSDL to the service registry (UDDI) for describing its service.

2) Service Registry/Broker

Service broker and service registry, these two names used interchangeably. It is responsible for providing details of published web services to any potential service requestor. It provides WSDL to the service requestor to invoked service directly. The Universal Description, Discovery and Integration (UDDI) specification defines a way to publish and discover information about Web services.

3) Service Requestor

The service requestor or client finds services in the service registry (UDDI) using various find operations and then binds to the service provider to invoke one of its Web services

3. Enterprise Application integration

Enterprise Application Integration is one of thrill word which is recycled all time in the software industries. Its name itself gives the basic idea about its functionality. EAI is a general category of approaches which provides interoperability between the multiple heterogeneous (disparate) systems that make up a typical enterprise infrastructure for communication. Basically, large organization is supports hundreds of applications. Most of these applications are legacy systems like SAP, Mainframe computers and mostly which is written in COBOL. An organization may be used separate-separate applications, for with their different functionalities. This modularization is always appropriate of large systems. It is easier for implementation to modularizing the task of running a business into multiple smaller tasks. New technologies are developed in very high speed by providing advantages on Legacy system. Therefore it is natural for organization to reduce legacy maintenance costs. The first approach is by replacing Legacy system with new system, but it is not good option to replace Legacy system. The second approach is instead of replacing Legacy system, build infrastructure between new system and existing Legacy system and we can do this using Enterprise Application Integration. However, to gain the advantages of such kind of distributed and modular system of an enterprise must implement technologies which deal with the following problems: Interoperability, Data integration, Robustness, Scalability and Stability.

Interoperability

Various components of the infrastructure can be used different platform, language, data formats, and may be different operating system, so we have to handle interoperability.

Data Integration

Flowing of data between applications may be having different formats so need to integrate data by converting to destination format of the system.

Robustness, Stability and Scalability

Because they are the glue that holds together a modular infrastructure, integration solutions must be highly robust, stable, and scalable.

4. Types of Integration

There are two kind of integration architecture for EAI as per Florence Lin 3C05-CW2 March 2005 document: Direct Point-to-point and middleware-based integration.

Point-to-point Integration

It is useful when dealing with few numbers of applications and we want good communication speed between applications. Point-to-point integration is shown in fig. 4. There are so many problems with point-to-point integration as applications increases then it becomes very difficult to implement and also difficult to maintain. It becomes tightly coupling between different applications. So that next type of integration such as Middleware based approach has been introduced

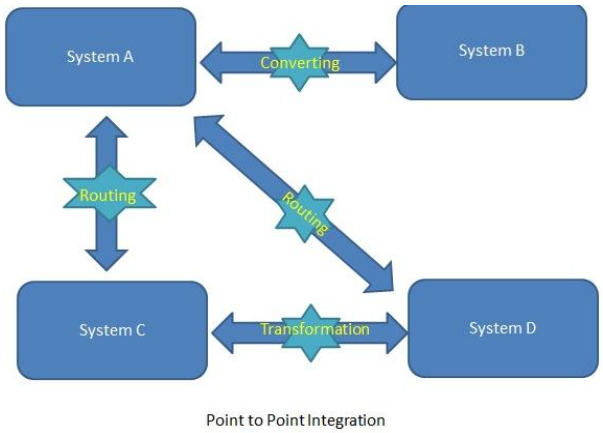

Fig. 4 Point-to-point Integration

Middleware based Integration

A middleware layer provide generic interface through which integrated (disparate or heterogeneous) systems are able to communicate with each other. Middleware layer will have provides functionalities to performs jobs such as transforming, routing, aggregating (Fan-out and Fan-in), converting and simple passing data. It forms loosely

coupled architecture. Middleware based integration, supports large amount of applications with minimum maintenance cost as compare to point-to-point integration. Middleware based integration is shown if fig. 5.

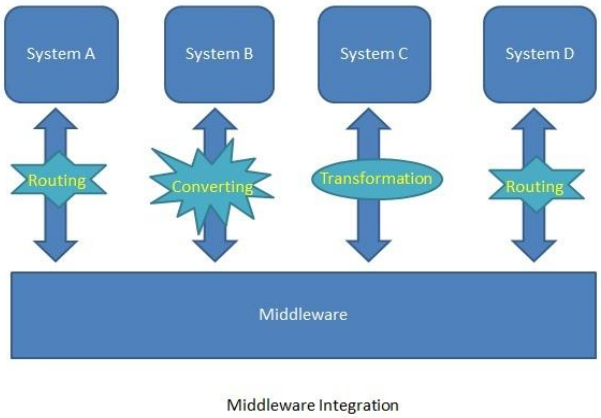

Fig. 5 Middleware based Integration

As per Florence Lin 3C05-CW2 March 2005 document, four types of integration for EAI: Data level integration, Application level integration, Method level integration and User interface level integration.

5. Literature Survey

The idea of enterprise application integration (EAI) and service oriented architecture (SOA) both are not new thing for software industry. We have gone through survey for EAI and SOA. There are many methods available to embrace the advantages new application with the old applications means legacy systems. First method can be possible to adopt of new application/database/technologies advantages and features are by replacing old database/application by new database/application/technology. This method has so many disadvantages because some business has been working from 10-15 years with legacy systems. It is complicated to move all the data in new systems. Another disadvantage is organizations already had invested huge amount of money in those systems.

Another method is to achieve to integrate enterprise application/database is using CORBA or J2EE. But this method also has so many disadvantages, one disadvantage is it develop layer between applications in tightly coupled architecture and it will becomes very complex to change in future because if we try to change one part of the application which will effect on another part also.

The next method is to integrate enterprise application using web services/Messaging Queue (SOA). This method has many advantages over the last two methods. First advantages is built integration layer between applications to communicate without replacing legacy system or old applications, so we can take advantages of both legacy system and new application. EAI using web service/Messaging Queue is creating infrastructure between applications for communication in loosely coupled manner so changes in one service which does not effect on another service because independent to another. If we will implement infrastructure using web service, still one issue has faced because of web service is time dependent. To remove this issue, the fourth method for enterprise application integration use both messaging queue and web services together.

6. Software Requirement Specification

Integration services that are well defined understood and managed which is the requirement of an open and interoperable between inside or outside enterprises. Following is integration services design principles:

- To achieve loosely coupled architecture between the applications common protocol and business semantics should be used.
- The service should be reusable across the electronic business means it will be unique unit of work.
- Service design should be driven by business requirements and reflected in the architecture
- Service design should be governed with a common approach and framework to achieve conceptual integrity.
- Services should be abstract, precise.
- Integration layer is used to achieve guaranteed delivery, managed integration.

7. Sample High Level Diagram

In this paper, we are going to provide two scenarios: First is Web Service based and second is Messaging Queue based. For better understanding will going to provide high level diagram for both scenario. Web service based scenario is shown in fig. 6 while Messaging Queue based scenario is shown in fig. 7

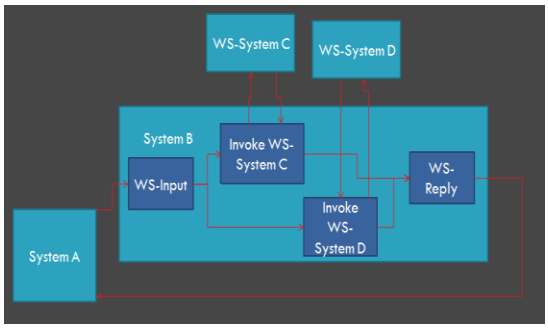

Fig. 6 Sample web service aggregation scenario

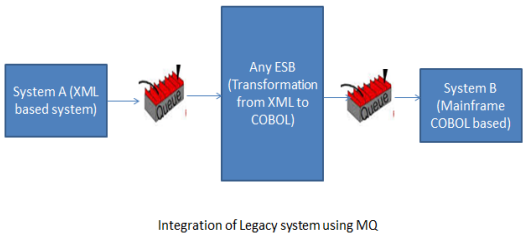

Fig. 7 Messaging Queue based

8. Generic Pattern

Generic pattern will avoid to creating the solution from scratch of commonly occurring problem. Suppose one

industry will have to develop around 100 interfaces and in which 80 interfaces is almost same or same scenario, and so instead of developing each interface from scratch, we will identify generic things between those interfaces. After identifying, we will develop pattern and whenever wants to develop the interface will just create instance of pattern and change as per requirements. It will provide huge amount of reusability in business.

Conclusion and future scope

In this paper we learned the role of Service oriented Architecture in Enterprise Application Integration. The electronic business has achieved the loosely coupled architecture directly or indirectly using SOA. The disparate systems will efficiently communicate with each other using service oriented architecture. For achieving this we have seen two scenarios in this paper, those scenarios are based on web service and message queue. Additions to these had provided generic pattern which is a reusable solution to a common problem within a given context. It will save time and money in future.

In next paper, we will provide detail implementation of systems very soon.

There are several issues that could be seen in future: Security aspects of web services and MQ, Queue monitoring, web service monitoring, and logging/Auditing messages etc.

Acknowledgment

This research paper has completed only because support from each and every one including: Government Polytechnic, Ahmednagar, Vishwabharati Academy College of Engineering, Ahmednagar, colleague, parents, friends, and my dear students.

Especially, my acknowledgment of gratitude toward the following important persons:

First, I would like to thanks to Government Polytechnic, Ahmednagar, MSBTE, Mumbai and DTE, Maharashtra.

Second I would like to thank Dr. Kureshi Sir Principal, Mrs. Sarika Joshi Madam Head of Computer Engineering, Mr. M. C. Kshirsagar Sir, Mr. Prabhudev sir, Mr. Natikar sir, Mr. P. C. Jaypal sir, and Mr. Ganesh Kadam sir Vishwbharati Academy College of Engineering, Ahmednagar to their support and encouragement

Last but not least, I sincerely thank to my parents and S. A. Bhalerao who provide the advice and financial support. This research paper will not be possible without all of them.

References

Victor R. Basili *et al*., Software Reuse: A Framework for Research unpublished.

Jitendra Joshi, Nisha Singh and Manjari Kumara. (2012). Web Service oriented Architecture Modeling with Pattern for Electronic Business Organization. IJARCSSE Research Paper [Online]. 2(9), pp. 456-460. Available www.ijarcsse.com/ docs/ papers/9_ September2012/.../V2I900179. pdf

Venkat N Gudivada and Jagadeesh Nandigam (2005), Enterprise Application Integration Using Extensible Web Services in Proc. of the IEEE ICWS'05.

Roshan Kulkarni (March 2006), Integration of Database in Service Oriented Architecture unpublished seminar report at IIT Bombay.

Michael N. Huhns and Munindar P. Singh. (2005). Service-oriented computing: Key concepts and principles. IEEE Internet Computing [Online]. 9(1), pp. 75-81. Available http://ieeexplore.ieee.org/ servlet/opac?punumber=4236

Mike P. Papazoglou (2003), Service Oriented Computing: Concepts, Characteristics and Directions, presented at the IEEE 4th Int. Conf. Web Information Systems Engineering.

Kostas Kontogiannis, *et al*. (2002), On the Role of Services in Enterprise Application Integration, presented at the IEEE 10th Int. workshop Software Technology and Engineering Practice.

Florence Lin (March 2005), Enterprise Application Integration (EAI) Techniques.

Jesus Bisbal *et al*., A Survey of Research into Legacy System Migration unpublished.

Matjaz B. Juric, Ivan Rozman and Marjan Heriko, 8) A method of integrating legacy system within distributed object architecture, unpublished

IBM (2008), Using Web Services with message flow in IBM WebSphere Message Broker V6.1 Developer Workshop, IBM, pp. 10.1-10.23.

Nilesh Vishwasrao Patil. (2014). Pattern Authoring: Web Service-oriented computing IJCET [Online]. 4(5), pp. 3143-3146. Available http://inpressco.com/wp-content/uploads/2014/09/Paper103143-3146.pdf

Chunhua Gu, Xueqin (2010), An SOA Based Enterprise Application Integration Approach in 2010 Third International Symposium on Electronic Commerce and Security of the IEEE.

Mohammed Said, Osama Ismail, and Hesham Hassan. (2013). Web Service Composition and Legacy Systems: A Survey IJCA [Online]. 69(16), pp. 9-15. Available

Website links:

- http://www.ibm.com/developerworks/websphere/library/techarticles/1010_stewart/1010_stewart.html
- http://www.tutorialspoint.com/webservices/web_services_characteristics.htm
- http://publib.boulder.ibm.com/infocenter/wsdoc400/v6r0/index.jsp?topic=/com.ibm.websphere.iseries.doc/info/ae/ae/cwbs_soawbs.html
- http://www.mulesoft.com/resources/esb/enterprise-application-integration-eai-and-esb
- http://www.w3.org/TR/wsdl
- http://www.tutorialspoint.com/wsdl/wsdl_elements.htm
- http://paulstovell.com/blog/integration/messaging

Biography

Nilesh Vishwasrao Patil is pursuing Master of Engineering with specialization in Computer Engineering from Vishwbharti Academy College of Engineering Ahmednagar. He has around five years of experience, in which around two years of industrial experience as Enterprise Application Integration (EAI) developer. Now he is working as System Analyst in Government Polytechnic, Ahmednagar since November 2011. He has published three international papers (Apache Hadoop – Resourceful Big Data Management, Dynamic Memory Allocation: Role in Memory Management and Pattern Authoring: Web Service Oriented Architecture) in reputed journal and presented one paper in National conference on Apache Hadoop – Resourceful Big Data Management.

M. C. Kshirsagar is working as assistant professor in Vishwbharti Academy College of Engineering Ahmednagar in Computer Engineering Department. He has completed Master of Engineering in Computer.

P. C. Jaypal is working as assistant professor in Vishwbharti Academy College of Engineering Ahmednagar in Information technology department. He has completed Master of Engineering in Computer.