P-3620

SERVICE COMPOSITION IN WIRELESS SENSOR NETWORKS

P-3620

A PROJECT REPORT

Submitted by

LIDIYA JOSEPH (0710108025)

SANDHYA.R (0710108042).

In partial fulfillment for the award of the degree of

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

KUMARAGURU COLLEGE OF TECHNOLOGY

(An Autonomous Institution Affiliated to Anna University of Technology, Coimbatore)

COIMBATORE – 641 049

APRIL 2011

KUMARAGURU COLLEGE OF TECHNOLOGY

(An Autonomous Institution Affiliated to Anna University of Technology, Coimbatore)

COIMBATORE - 641049

Certified that this project report entitled "**Service Composition In Wireless Sensor Networks**" is the bonafide work of "**Lidiya Joseph, Sandhya. R**" who carried out the research under my supervision.

SIGNATURE

[illegible]

Mrs.V.Vanitha

PROJECT GUIDE

Associate Professor

Department of Computer Science and Engineering

Kumaraguru College of Technology

Coimbatore-641049

SIGNATURE

[illegible]

Mrs.P.Devaki

HEAD OF THE DEPARTMENT

Department of Computer Science and Engineeri

Kumaraguru College of Technology

Coimbatore-641049

The candidates with **university Register Nos. 0710108025 & 0710108042** was examined by us in project viva-voce examination held on 20-04-2011.

[illegible]

INTERNAL EXAMINER

[illegible]

EXTERNAL EXAMINER

DECLARATION

We hereby declare that the project entitled " **Service Composition In Wireless Sensor Networks**" is a record of original work done by us and to the best of our knowledge, a similar work has not been submitted to Anna University of Technology or any Institutions, for fulfillment of the requirement of the course study.

The report is submitted in partial fulfillment of the requirement for the award of the Degree of Bachelor of Computer Science and Engineering of Anna University of Technology, Coimbatore.

[illegible]

Place: Coimbatore

(Lidiya Joseph)

Date: 20-04-2011

[illegible]

(Sandhya R)

ACKNOWLEDGEMENT

First and foremost, we would like to thank the **Lord Almighty** for enabling us to complete this project.

We extend our sincere thanks to our management, our chairman, **Dr.N.Mahalingam**, our co-chairman **Dr. Krishnaraj Vanavarayar**, and our director, **Dr.J.Shanmugam** for providing us with the necessary facilities and oppurtunities.

We extend our sincere thanks to our Principal, Dr.**S.Ramachandran,** Kumaraguru College of Technology, Coimbatore, for being a constant source of inspiration and providing us with the necessary facility to work on this project.

We would like to make a special acknowledgement and thanks to **Mrs.P.Devaki M.E.,(Ph.D).,** Head of Department of Computer Science &Engineering, for her support and encouragement throughout the project.

We express our deep gratitude and gratefulness to **our guide Mrs.V.Vanitha M.E.,** Department of Computer Science & Engineering, for her supervision, enduring patience, active involvement and guidance.

We would like to convey our honest thanks to our **class advisor, Mrs. C.Ramathilagam** and **all the other Faculty members** of the Department for their enthusiasm and wealth of experience from which we have greatly benefited.

We would also like to thank all our **non teaching staffs** for their constant support.

We also thank our **friends and family** who helped us to complete this project fruitfully.

ABSTRACT

A wireless sensor network is composed of large number of sensor nodes that are densely deployed either inside a phenomenon or very close to it. The main objective of the wireless sensor network is to observe an environment, collect information about the observed phenomena and events and deliver this information to the application.

Service Oriented Architecture has gained significant attention as programming model for heterogeneous wireless sensor networks. Its key idea is to exploit the decoupling of service providers and consumers to enable platform independent applications that are dynamically bound to platform specific services. The promise of service oriented architecture is to enable the composition of new distributed applications/solutions: when no available service can satisfy a client request, available services can be composed in order to satisfy such a request.

Dynamic service composition remains as one of the key challenges of service oriented computing today. In general, "service composition" can be defined as creating a composite service, obtained by combining available component services. In terms of software engineering automatic service composition will significantly enhance the power of service oriented architectures. Specifying manually which base services to use and how to combine them is cumbersome and ineffective with a rising number of relations between services and applications.

So we have implemented a simple approach to create on-demand process model for service composition in service oriented wireless sensor network, based on temporal dependency between services using a combination of forward and backward chaining algorithms. We have also generated the corresponding flow diagrams for better understanding. The experimental results show the increase in efficiency due to the composition of services.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF ABBREVIATIONS

SSI – Sensor Service Information Table

WSN-Wireless Sensor Networks

FC-Forward Chaining

BC-Backward Chaining

SOA-Service Oriented Architecture

SOWSN-Service Oriented Wireless Sensor Networks

WSCR - Web Service Composition Results

QoS - Quality of Service

LIST OF FIGURES

CHAPTER I

1. INTRODUCTION

Sensors integrated into structures, machinery, and the environment, coupled with the efficient delivery of sensed information, could provide tremendous benefits to society. Potential benefits include: fewer catastrophic failures, conservation of natural resources, improved manufacturing productivity, improved emergency response, and enhanced homeland security. However, barriers to the widespread use of sensors in structures and machines remain. Bundles of lead wires and fiber optic "tails" are subject to breakage and connector failures. Long wire bundles represent a significant installation and long term maintenance cost, limiting the number of sensors that may be deployed, and therefore reducing the overall quality of the data reported. Wireless sensing networks can eliminate these costs, easing installation and eliminating connectors.

1.1 WIRELESS SENSOR NETWORKS

Wireless Sensor Networks (WSN) are distributed networks of small sensors equipped with processors, memory and short range wireless communications. Conventionally, the main entities of WSN are sensors, aggregators, sink and gateway. Sensors do the actual sensing (i.e. detecting a phenomenon of interest), perform limited computation to decide where and how to send the data. They also act as routers if multi hop routing is used. The aggregators summarize (aggregate) data from the nearby sensors within a region of interest. These could be specialized nodes or dynamically selected from among the sensors. The sink is a collector of data from all the sensors and/or aggregators. Sensors, aggregators and the sink usually communicate via proprietary radio links and protocols. The gateway is the one that bridges the WSN to the 'outside world'. It is equipped with dual network interface to be able to communicate with the sink (and sensors) as well as with end user application devices or the external infrastructure and provide the necessary mappings and protocol conversions. The number of nodes in a Sensor Network can be several orders of magnitude higher than that in ad hoc network. Sensor nodes are densely deployed, and are prone to failures. Deployment scenarios for sensor networks are countless and diverse. For example, sensors may be used for military applications, weather forecasting, tsunami detection, pollution detection, power management in schools and office buildings and so on.

Until now, research in the wireless sensor network (WSN) domain has mainly focused on routing, data aggregation, and energy conservation inside a single sensor network while the integration of multiple sensor networks has only been studied to a limited extent. However, as the price of wireless sensors diminishes rapidly we can soon expect large numbers of autonomous sensor networks being deployed. These sensor networks will be managed by different organizations but the interconnection of their infrastructures along with data integration and distributed query processing will soon become an issue to fully exploit the potential of this "Sensor Internet." This requires platforms which enable the dynamic integration and management of sensor networks and the produced data streams.

1.2 MOTIVATING APPLICATIONS

WSNs can be used for cargo tracking and monitoring by attaching sensors to individual cargo containers. However, containers are frequently too far apart to be covered by a single WSN, since they are housed in separate warehouses and eventually relocated by boat or rail. Thus, the sensors form multiple independent WSNs which are unable to directly communicate with each other. The utility of the cargo tracking application would greatly increase if the user could issue a query such as searching the containers for a specific item simultaneously to all of these containers, even though their WSNs are not physically connected.

Building monitoring and control

Building monitoring and control system is normally required to perform:

• Indoor environmental monitoring (heating, ventilation and air conditioning services). An air conditioner acts as an actor device that periodically receives measurements from temperature and humidity sensors. The air conditioner adjusts the air quality to meet user preferences

• Response to extreme events such as fire. When a water sprinkler device sporadically receives a high temperature event, it asks for the smoke level. If smoke detectors actually report the presence of fire, the water sprinkler is operated and an event is sent to a fire alarm control device, which can then activate emergency bells, send an emergency message to some subscribed target (e.g. the PDA of a fire fighter) and activate some other nearby water sprinklers.

•Structural monitoring. A vibration control device periodically receives measurements from accelerometer sensors so that possible vibrations suffered by the building can be controlled.

• Security control. Sporadic noise level events can be sent from the corresponding sensors to actor devices able to activate cameras and send messages to centralized control devices.

Emergency Medical Services

A sensor infrastructure can incorporate wireless vital sign sensors (*e.g.,* pulse oximetry, EKG, respiration) attached to patients; real-time sensors of traffic conditions and ambulance location through GPS; hospital and availability of beds in the emergency room; and the status of specialized facilities such as trauma and burn centres. By allowing this information to be queried, filtered, and relayed to display terminals, handheld computers, or laptops carried in an ambulance or in the hospital, health care providers can obtain a real-time view of the status of patients, location of ambulances, and the status of medical facilities.

Parking-space finder

The parking-space finder uses cameras throughout a metropolitan area to track parking space availability. Users fill out a Web form to specify a destination and any constraints on a desired parking space (for example, does not require a permit, must be covered, and so on). Based on the input criteria, the parking space- finder service identifies the nearest available parking space that satisfies the user constraints, then uses the Yahoo! Maps service to find driving directions to that parking space from the user's current location.

Coastal imaging service

The service uses cameras installed at sites along the Oregon coastline and processes the live feed from the cameras to identify the visible signatures of near shore phenomena such as riptides, sandbar formations, and so on.

Other applications

- Automobile navigation systems could become aware of traffic conditions, weather, and road conditions along a projected route.
- Network and host monitor service collects data from host and network monitoring tools (which act as sensing devices) running on a widely dispersed set of hosts and lets users query those data efficiently.

- A retailer with several outlets at shopping malls across a nation could access video streams from security cameras at those malls, the retailer could build custom business applications for understanding customer behaviour
- Imaging service for tourist sites where cameras are installed. These cameras can form a WSN and provide some pictures or video streams, and allow remote users to query their interested information.

Area monitoring

Area monitoring is a common application of WSNs. In area monitoring, the WSN is deployed over a region where some phenomenon is to be monitored. A military example is the use of sensors to detect enemy intrusion; a civilian example is the geo-fencing of gas or oil pipelines.

When the sensors detect the event being monitored (heat, pressure), the event is reported to one of the base stations, which then takes appropriate action (e.g., send a message on the internet or to a satellite). Similarly, wireless sensor networks can use a range of sensors to detect the presence of vehicles ranging from motorcycles to train cars.

Environmental monitoring

A number of WSNs have been deployed for environmental monitoring

Air pollution monitoring

Wireless sensor networks have been deployed in several cities (Stockholm, London or Brisbane) to monitor the concentration of dangerous gases for citizens. The sensor nodes can control important parameters like CO, CO2, NO2 or CH4, which are generated by vehicles and industry, and have a severe impact on the human health. This way, the public institutions have a good tool to design plans to reduce pollution, improve the air quality and ensure the compliance with current legislation.

Forest fires detection

A network of Sensor Nodes can be installed in a forest to control when a fire has started. The nodes will be equipped with sensors to control temperature, humidity and gases which are produced by fire in the trees or vegetation. The early detection is crucial for a successful action of the fire fighters; thanks to Wireless Sensor Networks, the fire brigade will be able to know when a fire is started and how it is spreading.

Greenhouse monitoring

Wireless sensor networks are also used to control the temperature and humidity levels inside commercial greenhouses. When the temperature and humidity drops below specific levels, the greenhouse manager must be notified via e-mail or cell phone text message, or host systems can trigger misting systems, open vents, turn on fans, or control a wide variety of system responses.

Landslide detection

A landslide detection system makes use of a wireless sensor network to detect the slight movements of soil and changes in various parameters that may occur before or during a landslide. And through the data gathered it may be possible to know the occurrence of landslides long before it actually happens.

Industrial monitoring

Machine health monitoring

Wireless sensor networks have been developed for machinery condition-based maintenance (CBM) as they offer significant cost savings and enable new functionalities. In wired systems, the installation of enough sensors is often limited by the cost of wiring. Previously inaccessible locations, rotating machinery, hazardous or restricted areas, and mobile assets can now be reached with wireless sensors.

Water/Wastewater monitoring

There are many opportunities for using wireless sensor networks within the water/wastewater industries. Facilities not wired for power or data transmission can be monitored using industrial wireless I/O devices and sensors powered using solar panels or battery packs.

Landfill ground well level monitoring and pump counter

Wireless sensor networks can be used to measure and monitor the water levels within all ground wells in the landfill site and monitor leachate accumulation and removal. A wireless device and submersible pressure transmitter monitors the leachate level. The sensor information is wirelessly transmitted to a central data logging system to store the level data, perform calculations, or notify personnel when a service vehicle is needed at a specific well.

Agriculture

Using wireless sensor networks within the agricultural industry is increasingly common; using a wireless network frees the farmer from the maintenance of wiring in a difficult environment. Gravity feed water systems can be monitored using pressure transmitters to monitor water tank levels, pumps can be controlled using wireless I/O devices, and water use can be measured and wirelessly transmitted back to a central control center for billing. Irrigation automation enables more efficient water use and reduces waste.

Fleet monitoring

It is possible to put a node on-board of each vehicle of a fleet. The node gathers its position via the GPS module, and reports its coordinates so that the location is tracked in real-time. The nodes can be connected to temperature sensors to avoid any disruption of the cold chain, helping to ensure the safety of food, pharmaceutical and chemical shipments. In situations where there is not reliable GPS coverage, like inside buildings, garages and tunnels, using information from GSM cells is an alternative for to GPS localization.

1.3 BENEFITS OF WIRELESS SENSOR NETWORKS

There are many advantages of wireless sensor networking some of important prose's are:

- They have no hassle of cables
- It can work efficiently under the harsh conditions
- It can be deployed up to large scale.
- It can accommodate new devices at any time
- Its flexible to go through physical partitions
- It can be accessed through a centralized monitor

1.4 CHALLENGING ISSUES IN WIRELESS SENSOR NETWORKS

Wireless sensor networks (WSN) bring out many new challenging issues. In this section, we try to articulate these challenges.

Heterogeneity

Components of a WWSN are highly heterogeneous along two dimensions. In the sensor node dimension, the types of sensors (small embedded devices, mobile phones, wireless cameras), and the capabilities for processing data are different. In the sensor network dimension, the number of nodes and the rate for producing data are different. These heterogeneities demand a standardized abstraction to achieve a better understanding of different sensor information and a uniform solution for gathering different sensor data.

Scalability

A WWSN becomes more useful as the number of participant WSNs grows. Aiming at a very large number of data producers and consumers with a variety of application requirements, this introduces a significant challenge for scalability. To keep the resource usage scalable and to avoid unnecessarily denying access to more applications, it's essential for the system to reuse common data and sensing resources among overlapping application needs. This scalability challenge is further compounded by the need for maintaining extensibility to unknown applications and domain-specific data processing & aggregation.

Publishing and discovering sensor resources

If publishing even a single stream of sensing data requires too much effort, this will discourage data owners to contribute sensing data. So, the effort requested for publishing sensing data is a significant issue, which has direct impact on participant numbers. Not only is the publishing very important, but also the discovering sensor data is a significant issue. The shared WWSN should have the ability to discover and invoke services from heterogeneous WSNs and fuse related data to cater for different applications.

Query aggregation

There will be a large number of applications over this WWSN, although different applications may seek different sensor data, a subset of data collection tasks is common to some services. Wasting computation and limited energy resource should be avoided if two sensing applications overlap. An arbitrary number of user queries can be merged into a single query, which can avoid the overlapping queries on the same data source and decrease the processing load in the gateway. Therefore, how to aggregate a large number of queries, optimize the processing load in the gateway and efficiently share sensing streams among multiple applications is a big challenge issue.

Interconnection

To achieve sensing resources sharing in the WWSN, an appropriate interconnection approach must be introduced to answer: how to interconnect different sensor networks which are spatially deployed in different locations with IP based Internet; how will the designed interconnection solution support the integration issues in both network layer and data layer. The design of this interconnection solution will have strong impact on all the solutions for other issues, which actually requests the designers to take the cross-layer interaction into consideration.

Integration

Integration follows the challenge of interconnection. Sensor networks which are physically located in different locations may use totally different routing protocols for their specific applications. How to integrate these heterogeneous WSNs into one shared system over Internet to provide comprehensive services for users is the critical issue after connecting sensor networks to IP based Internet.

Data collection and data storage

The system employs numerous globally distributed sensors that observe the physical world. Because the sensors would collect a vast volume of data, and because we would want to retain both the most recent observations and a historical record, the system should collect observations and store them. How to efficiently collect and store global sensor data is a critical issue.

APIs for high-level applications

As the WSN participant number grows, a variety of new kinds of applications over existing data networks can be deployed by third party. Appropriated open APIs should be provided to ease the development of these kinds of applications. This programming paradigm can conversely incentive more data producers and consumers to participate.

1.5 SERVICE ORIENTED WIRELESS SENSOR NETWORKS

Sensor webs are heterogeneous collections of sensor devices that collect information and interact with the environment. They consist of wireless sensor networks that are ensembles of small, smart and cheap sensing and computing devices that permeate the environment as well as high bandwidth rich sensors such as satellite imaging systems. While internet protocols and web standards provide well. developed mechanisms for accessing this information, linking such mechanisms with response-constrained sensor networks is very challenging because of the volatility of the communication links. This implementation utilizes a service oriented programming model for sensor networks which permits discovery and access of web services.

The term service oriented architecture refers to a logical set that consists of several large software components that together perform a certain task or service. SOA is a particularly paradigm in the community of the web software developers, for example web services utilize this architecture. The SOA allows programmers to access wireless sensor networks from their applications by using a simple service-oriented API via the language of their choice. The service oriented approach provides adequate abstractions for application developers, and that is a good way to integrate the internet with WSN.

As opposed to established WSNs which are tightly coupled with specific applications, service oriented computing paradigm enables flexible composition of various applications using multiple reusable services. A service oriented architecture offers flexibility in the design of WSN applications since it provides accepted standards for representing and packaging data, describing the functionality of services, and facilitating the search for available services which can be invoked to meet application requirements.

1.6 ADVANTAGES OF SERVICE – ORIENTED ARCHITECTURE

- It is platform independent and programming language independent.
- Location transparency
- Code reuse is possible
- Greater testability
- Parallel development of services is possible
- Better scalability
- Higher availability

- Dynamic development of services – new services can be deployed on demand or existing ones can be updated.

1.7 SERVICE COMPOSITION IN WIRELESS SENSOR NETWORKS

Wireless sensor network (WSN) service composition has been recently proposed as a method to rapidly develop applications in WSNs. In WSNs, a query task may require a set of services and may be carried out repetitively with a given frequency during its lifetime. A service composition solution shall be provided for each execution of such a persistent query task. Due to the energy saving strategy, some sensors may be scheduled to be in sleep mode periodically. Thus, a service composition solution may not always be valid during the lifetime of a persistent query.

In many cases, a single service stored in any of the nodes of the WSN may not be able to satisfy the user's request. To overcome this drawback in WSN, it is not optimal to populate the nodes with more and more services. Hence, if a single service cannot answer the query, then dynamic discovery and composition of the abstract candidate services has to be done and this, in most of the cases, satisfies the user's request and thus increases the efficiency of Wireless Sensor Nodes.

Dynamic service composition remains one of the key challenges of service oriented computing today. In general, "service composition" can be defined as creating a composite service, obtained by combining available component services. It is used in situations where a client request cannot be satisfied by any single available service, but by a combination thereof. In terms of software engineering automatic service composition will significantly enhance the power of service oriented architectures. SOAs combine available base services in order to build higher level services or distributed applications. Specifying manually which base services to use and how to combine them is cumbersome and ineffective with a rising number of relations between services and applications. Furthermore the resulting composite process is vulnerable to change as it must be corrected manually when base services become unavailable or new better services appears.

Traditionally, the task of automatic service composition has been split into four phases:

(1) Planning (Process model creation)

(2) Discovery

P-3620

(3) Selection and binding

(4) Execution

The first phase involves generating a plan, i.e., all the services and the order in which they are to be composed in order to obtain the composition. The plan may be generated either statically or dynamically. In WSN dynamic composition techniques are getting more preference due to their potential for handling unpredictable changes of runtime environment that cannot be handled using static composition techniques.

The second phase involves discovering services as per the plan. Depending on the approach, often planning and discovery are combined into one step. After all the appropriate services are discovered, the selection phase involves selecting the optimal solution based on non-functional properties like QoS properties.

The last phase involves executing the services as per the plan and in case any of them are not available, an alternate solution has to be used.

The project deals only with the planning phase of service composition and its key contributors are as mentioned above in the problem definition.

1.8 AN EXAMPLE OF SERVICE COMPOSITION

Service composition is useful when we are looking for a web service with specific input and output and there is no single service which satisfies the request. For example, assume that a user wants to find the service whose input is "City" and output is "Weather" among services listed in the Table 1.1. If the service composition is not supported, there is no answer to the user query. However, if the composition is supported, a sequence of service can be provided as the answer to the user query. The composition of services S1 and S2, in which the output of S1 is equal to the input of S2, satisfies the user query. Similarly for the user query with input City and output SportsOK the sequence of services are S1, S2, and S4.

ID	Name	Operation	Input	Output
S1	PlaceLookup	CitytoZipcode	City	ZipCode
S2	WeatherByZipcode	WeatherInfo	ZipCode	Weather
S3	TemperatureFetcher	GetTemperature	Postcode	Temperature
S4	SportsStatusViewer	SportsByWeather	Weather	SportsOK

Table1.1 - Example services

This ability of composing service using multiple services allows meeting larger and single user requirements that could not otherwise be met with any of the available smaller services. Thus complex service based applications can be created by composing individual services.

Usually services that are created by same or different providers are meant to be accessed and made to work independent to each other. But establishment of composite service based applications necessitates interaction, communication, cooperation and communication of services. This necessitates creation of a process model.

Composition of 2 Services

User query: given input as city, output required is weather

Process Model: S1, S2

By composing the two services CitytoZipcode and WeatherInfo, the user request gets satisfied.

Composition of 3 Services

User query: given input as city, output required is SportsOK

Process Model: S1, S2, S4

As a single service does not satisfy the request, three candidate services CitytoZipcode, WeatherInfo and SportsStatusViewer are composed to satisfy the user request.

CHAPTER II

2. LITERATURE SURVEY

2.1 RELATED WORKS ON WIRELESS SENSOR NETWOKS

In [2], the concept of sensor networks which has been made viable by the convergence of micro-electro-mechanical systems technology, wireless communications and digital electronics is described. First, the sensing tasks and the potential sensor networks applications are explored, and a review of factors influencing the design of sensor networks is provided. Then, the communication architecture for sensor networks is outlined, and the algorithms and protocols developed for each layer in the literature are explored. Open research issues for the realization of sensor networks are also discussed.

In [3], the work done is to bridge the gap between high-end networked devices and wireless networks of ubiquituous and resource-constrained sensors and actuators by extensively applying Service-Oriented Architecture (SOA) patterns. The authors have presented a multi-level approach that implements existing SOA standards on higher tiers, and propose a novel protocol stack, WSN-SOA, which brings the benefits of SOA to low capacity nodes without the overhead of XML-based technologies. The solution fully supports network dynamicity, auto-configuration, service discovery, device heterogeneity and interoperability with legacy architectures. As a proof-of-concept, they have also studied a surveillance scenario in which the detection of an intruder, conducted within the range of a network of wireless sensors (e.g., MICAz from Crossbow), leads to the automatic triggering of tracking activities by a Linux-powered network camera and of alerts and video streams toward a control room.

In [6], it is studied that the characteristics of wireless sensor networks (WSNs) make their management different from traditional networks. With service oriented concepts having been implemented in WSNs, the paper investigates service oriented management of heterogeneous WSNs. A services oriented management architecture for heterogeneous WSNs is proposed in order to make better use of network management business processes from the point of view of WSNs managers/users. Within the proposed management architecture, entities and common management services are specified with heterogeneity of WSNs hidden by loose-coupling service oriented Web

Services technologies. Functional architectures of entities and key issues for implementing the prototype system are presented.

2.2 RELATED WORKS ON SERVICE COMPOSITION

In [1], given a set of candidate web services and a user's request description in terms of (I, O, P, E, G), the proposed method can find a composite service that would satisfy user's requirements in two steps. First, it anticipates the potential direct and indirect dependency between abstract services, and second, it generates process model (PM) automatically using the dependency information. The architecture and application of this method and its application are discussed using a case study. Moreover, a summary of existing techniques and their shortcomings are presented. This approach takes advantages of a sorting algorithm and semantic I/O matching techniques.

In [4], a data structure called WS chain table, which describes the dependent relations among Web services, is proposed, and an efficient algorithm for WS composition is developed based on the data structure. The time complexity of the new algorithm is O(max(log m,n)), improved the known O(n times log m) one, where n is the number of WSs and m denotes the number of output parameters of WSs. Experiments are carried out in order to compare the performance of the algorithms with different data structure.

In [5], a composition model capable of composing Web services across wide area networks with the service composition based on interface idea integrated with Peer to Peer technologies is presented. It forms a novel Web Services Composition Overlay Network (WSCON) with Peer to Peer technologies, then associate nodes in the same Web services composition domain to form the Web Services Composition Network, according to domain ontology and its reasoning ability. Web service discovery and composition entry are performed through Distributed Hash Table (DHT). They have conducted extensive experiments using large-scale simulations. The experiment results show the advantages of our model: supporting Web service composition based on QoS, high composition success rate, fast service distribution, discovery and composition, and tolerant to dynamic service component node arrivals and departures.

In [6], service composition is done using persistent queries. In WSNs, a query task may require a set of services and may be carried out repetitively with a given frequency during its lifetime. A service composition solution shall be provided for each execution of such a persistent query task. Due to the energy saving strategy, some sensors may be scheduled to be in sleep mode periodically. Thus, a service composition solution may not always be valid during the lifetime of a persistent query. When a query task needs to be conducted over a new service composition solution, a routing update procedure is involved which consumes energy. A service composition design is studied which minimizes the number of service composition solutions during the lifetime of a persistent query. They have also aimed to minimize the total service composition cost when the minimum number of required service composition solutions is derived. A greedy algorithm and a dynamic programming algorithm are proposed to complete these two objectives respectively. The optimality of both algorithms provides the service composition solutions for a persistent query with minimum energy consumption.

In [8], the mutual search operations among Web Service operations, inputs and outputs are studied, and a novel data structure called Double Parameter Inverted File (DuoParaInvertedFile) is proposed to implement these operations. The paper is based on the fact that the Web Service Composition Tree (WSCT) is the key that affects the performance of Web Services compositions when a large number of services are available An algorithm to build DuoParaInvertedFile is provided as well. Based on above results an efficient algorithm for constructing the WSCT is given. Furthermore, the QoS (Quality of Service) of Web Service Composition Results (WSCRs) is defined, and then an algorithm used to get the best WSCR in terms of QoS is proposed. Experiments show that this approach not only ensures the success of the performance of Web Services composition but also ensures the QoS of Web Services composition.

CHAPTER III

3. PROJECT DESCRIPTION

3.1 PROBLEM DEFINITION

Given a user query request for a service from wireless sensor networks which cannot be satisfied by a single candidate service, then two or more services are composed to satisfy the user request. The main objective of our project is to perform the planning phase of service composition using an efficient algorithm which includes:

- Discover candidate abstract services
- Find out the dependency between the abstract services, and
- Generate the process model automatically

A combination of forward chaining and backward chaining algorithm is used to efficiently generate the process model.

3.2 THE SENSOR SERVICE INFORMATION (SSI) TABLE

In wireless sensor networks each base station maintains a special data structure called sensor service information (SSI) table. This table describes the dependent relations among WSN services. Table consists of unique identifier of the service, name of the service, input parameter required for the service, output parameter produced by the service. Each service takes multiple inputs and produces multiple outputs. Tag field specifies whether the service is an independent service (0) or composed service (1). If tag field contains '0' then link field points to all services whose output is equal to the input of this service(used for backward chaining) and it also points to another list of all services whose input is equal to the output of this service(used for forward chaining). On the other hand if the tag field contains '1' , then link field points to list of services required to compose the service in their correct order. The structure of the SSI table is given in the Table3.1.

ID	Name	Input parameter	Output parameter	Tag	Link

Table 3.1 The structure of the SSI table

3.3 CANDIDATE ABSTRACT SERVICE DISCOVERY

The domain considered for our project is E-Marketing. Email marketing is a form of direct marketing which uses electronic mail as a means of communicating commercial or fund-raising messages to an audience. In its broadest sense, every email sent to a potential or current customer could be considered email marketing. However, the term is usually used to refer to sending email messages with the purpose of enhancing the relationship of a merchant with its current or previous customers, to encourage customer loyalty and repeat business, sending email messages with the purpose of acquiring new customers or convincing current customers to purchase something immediately, adding advertisements to email messages sent by other companies to their customers, and sending email messages over the Internet, as email did and does exist outside the Internet (e.g., network email and FIDO).

We have collected 100 services candidate abstract services in E-Marketing domain which are categorized into 6 categories. The categories are as follows:

- E-mail and boomerang services
- Sales and Management services
- ATM and Credit card services
- Information services
- Marketing services
- Movie services

Now suppose a user request UR (I, O) arrives, the algorithm searches the table for a service with output = O. From thereon it follows a link to find all the candidate services.

3.4 DEPENDENCY LIST COMPUTATION

The dependency list is computed dynamically for all the services of the SSI table. Two dependency lists are created, one for forward chaining and another for backward chaining.

3.4.1 DEPENDENCY LIST FOR BACKWARD CHAINING

Whenever a new service is added to the table, the algorithm checks whether output of this new service is input for any of the existing service. If so the new service is added to the dep_bck list which is pointed by the link field of the existing service. Similarly the algorithm checks whether input of this new service is output of any other existing service. If so the existing service is added to the dep_bck list of this new service.

```
For each entry E in SSI
For all the existing entries EE in SSI
if (E.inp == EE.outp)
      E.dep_bck.Add (EE.id)
if (E.outp == EE.inp)
      EE.dep_bck.Add (E.id)
```

3.4.2 DEPENDENCY LIST FOR FORWARD CHAINING

Whenever a new service is added to the table, the algorithm checks whether input of this new service is output for any of the existing service. If so the new service is added to the dep_fwd list which is pointed by the link field of the existing service. Similarly the algorithm checks whether output of this new service is input of any other existing service. If so the existing service is added to the dep_fwd list of this new service.

```
For each entry E in SSI
For all the existing entries EE in SSI
```

```
    if (E.inp == EE.outp)
        EE.dep_fwd.Add (E.id)
  if (E.outp == E.inp)
        E.dep_fwd.Add (EE.id)
```

3.5 PROCESS MODEL GENERATION

. We have implemented the backward chaining algorithm and then a combination of forward chaining and backward chaining to compare the efficiency of both methods.

Suppose a user request UR (I, O) arrives, the backward chaining algorithm searches the table for a service with output = O. From thereon it follows the link using the dep_bck list to find all the candidate services until the input matches with user input I. While combining both forward and backward chaining first when the user request arrives ,the algorithm searches the table for a service with output=O and then traverses the dep_bck list ,simultaneously the algorithm searches the table for a service with input=I and traverses the dep_fwd list .The traversed list web services are stored in separate arrays for forward and backward chaining .A parallel check is made to see whether the elements of the array are equal, when it is equal the traversal is stopped and the elements of the forward chaining array and backward chaining array are merged to give the process model. This method is more efficient than backward chaining method.

Once process model is created, in order to reuse the result and to reduce the time of the search process, the composite service is added to the SSI table with tag='1' along with the all the required services in the correct order.

A flow diagram is generated dynamically for a better understanding of the flow of inputs and outputs through services during the composition. We have taken two models into consideration.

1. *Sequential model*

In sequential model the flow is sequential i.e. the services are executed one after the other.

2. *Parallel model*

In parallel model, two or more services are executed simultaneously to generate the required output

3.6 ALGORITHM

Algorithm: generation of process model

Input: Sensor service information table, user request (UR)

Output: The service/services required to satisfy UR –The Process Model

```
    For each entry E in SSI
    For all the existing entries EE in SSI
    /* backward chaining dependency list computation*/
    if (E.inp == EE.outp)
            E.dep_bck.Add (EE.id)
  if (E.outp == EE.inp)
           EE.dep_bck.Add (E.id)
    /* forward chaining dependency list computation*/
    if (E.inp == EE.outp)
       EE.dep_fwd.Add (E.id)
   if (E.outp == E.inp)
       E.dep_fwd.Add (EE.id)
    /* process model generation*/
    if(E.inp==U.inp && E.outp==U.outp && E.tag==1)
            return E.link
    else if( E.inp==U.inp)
```

```
        if(E.outp==U.outp)
                return E.id
else  if(E.outp==U.outp)
        Push(E) to stack S
        traverse the dep_bck list
        for each service W in dep_bck list
L1:                 if W.inp==U.inp
                            Push(W) to stack S
                    else
                            traverse dep_bck of W
                                    for each service W1 of dep_bck ofW
                                            W=W1
                                            Goto L1
Else if (E.inp==U.inp)
        Push(E) to stack SF
        traverse the dep_fwd list
        for each service W in dep_fwd list
        L2:     if (W==Pop(S))
                        return(Pop(SF)+Pop(S))
                else
        traverse dep_fwd of W
                        for each service W1 of dep_fwd ofW
                                W=W1
                                Goto L2
Else
        User request cannot be satisfied
        Return
```

3.7 IMPLEMENTATION SPECIFICATIONS

Language used	:C#
Platform	: Microsoft visual studio 2008
Implementation Environment	: Visual studio 2008
Framework	: .NET framework 3.5
Operating System used	: Windows XP
Database	: Oracle 10g
Server	: ASP.NET development server

CHAPTER IV

4. IMPLEMENTATION RESULTS AND DISCUSSIONS

4.1 ALGORITHM ANALYSIS

Initially in the algorithm, only backward chaining was used to generate the process model. i.e., first the output of the user query is determined and the request_output(output category of the user request) is checked with the output field of the database. When a match is found, the dependency list of that service (whose output matched the request_output) is traversed and the process model was generated.

Later, in the algorithm, forward chaining is combined along with the backward chaining i.e. simultaneously the dependency list of both the services (the service whose output matched the request_output and the service whose input matched the request_input) are traversed.

But comparing the two algorithms, i.e. the one using only backward chaining and the one combining backward chaining with forward chaining, it is found that the response time decreased considerably.

The **time complexity** of the algorithm is **$O(n^2)$.**

4.2 IMPLEMENTATION RESULTS

Average response time with backward chaining alone = 1.222356 ms

Average response time with simultaneous forward
and backward chaining = 0.886231 ms

4.3 EXPERIMENTAL INFERENCE

Thus it is inferred that the performance has increased while combining forward chaining along with the backward chaining and the average response time has decreased by 0.336125 ms.

4.3.1 Percentage of improvement

Therefore, the percentage of improvement by combining forward chaining with backward chaining **= 0.336125/1.222356 * 100 %**

= 27.4981 %

4.4 EXPERIMENTAL RESULTS

The algorithm is implemented in .NET environment. Services in E-Marketing domain are considered and the service composition algorithm is applied on the services in this domain based on the user request. 100 services in E-Marketing domain are collected on which the algorithm is applied. The services are as listed in the following pages in Table 4.1.

To prove the performance increase when service composition is applied on these services, we have considered two set of user queries (set I and set II). Table 4.2 and Table 4.3 show the two sets.

For each set, performance graph is plotted, which shows success rate for both the cases (with service composition and without service composition)

Figure 4.1: performance graph for Set 1

Figure 4.2: performance graph for Set 2

ID	NAME	INPUT	OUTPUT	TAG	LINK
ws01	shoppingcart_service1	web_based_solution(purpose)	name of the tool	0	-
ws02	shoppingcart_service2	name of the tool	URL of the tool	0	-
ws03	corporate info service1	request(work to be done)	list of companies providing the solution	0	-
ws04	corporate info service2	company name providing the solution	company profile	0	-
ws05	corporate info service3	company name	works done	0	-
ws06	corporate info service4	logo designing options	logo designing templates	0	-
ws07	corporate info service5	business card designing options	business card templates	0	-
ws08	corporate info service6	letter head designing options	letter templates	0	-
ws09	corporate info service7	logo templates	range of eost	0	-
ws10	corporate info service8	business card templates	range of cost	0	-
ws11	corporate info service9	letter templates	range of cost	0	-
ws12	corporate info service10	envelope templates	range of cost	0	-
ws13	corporate info service11	envelope deigning options	envelope templates	0	-
ws14	corporate info service12	brochure designing options	brochure templates	0	-
ws15	corporate info service13	brochure templates	range of cost	0	-
ws16	shipment management service1	transportation type	load	0	-
ws17	shipment management service2	load	estimate cost	0	-
ws18	shipment management service3	route	transportation type	0	-
ws19	product management service1	product name	product id	0	-
ws20	product management service2	product id	use(purpose)	0	-
ws21	product management service3	product id	rating	0	-

ws22	product management service4	product id	cost	0	-
ws23	product management service5	product id	URL for download	0	-
ws24	product management service6	product id	user review	0	-
ws25	shipment management service4	Transportation type, distance	Cost of travel	0	-
ws26	strikeiron service1	country name	country code	0	-
ws27	strikeiron service2	country code	sales tax percentage for different range of amounts	0	-
ws28	strikeiron service3	country code,principle amount	amount after tax reduction	0	-
ws29	theatre and movie services	city name,zip code	theatre names	0	-
ws30	theatre and movie services2	theatre names	current movies	0	-
ws31	theatre and movie services3	current movies	show time	0	-
ws32	theatre and movie services4	theatre name, show time	availability of tickets	0	-
ws33	constant contact service1	business type	newsletter template	0	-
ws34	constant contact service2	newsletter template	Cost of template	0	-
ws35	ATM location service1	city name	city zip	0	-
ws36	ATM location service2	city zip	latitude, longitudinal location	0	-
ws37	ATM location service3	city zip	location	0	-
ws38	xignitc currency service1	country name	country code	0	-
ws39	xignite currency service2	country code	foreign exchange rate	0	-
ws40	xignite metal service1	country name	country core	0	-
ws41	xignite metal service2	country code	rate of metals available	0	-
ws42	credit card validator service1	card number	card type	0	-
ws43	credit card validator	card number	validity	0	-

	service2				
ws44	feedBlitz service1	blog id	online survey	0	-
ws45	feedBlitz service2	online survey	rating among blogs	0	-
ws46	feedBlitz service3	blog id	number of followers	0	-
ws47	middle east marketing service1	city name	zip code	0	-
ws48	middle east marketing service2	zip code	list of jeweller shops	0	-
ws49	middle east marketing service3	list of jeweller shops	owner name	0	-
ws50	middle east marketting service4	list of jeweller shops	location of the shop	0	-
ws51	iContact service1	icontact id	coupon no	0	-
ws52	iContact service2	coupon no	features	0	-
ws53	iContact service3	features	price for the feature	0	-
ws54	iContact service4	features	existing customers	0	-
ws55	iContact service5	existing customers	renewal options	0	-
ws56	iContact service6	icontact id	campaign options	0	-
ws57	iContact service7	campaign options	cost of the product	0	-
ws58	pinpointer service1	pinpointer user id	subscription options	0	-
ws59	pinpointer service2	subscription options	price	0	-
ws60	pinpointer service3	pinpointer id	bulk mail options	0	-
ws61	pinpointer service4	bulk mail options	existing customer email ids	0	-
wa62	pinpointer service5	pinpointer id	FAQs	0	-
ws63	pinpointer service6	FAQs	answers	0	-
ws64	pinpointer service7	pinpointer id	online training sections	0	-
ws65	pinpointer service8	online training sections	training duration, training cost	0	-
ws66	vertical payment service1	VP id	test drive options	0	-
ws67	vertical payment service2	test drive id	test drive features	0	-
ws68	vertical payment service3	VP id	vp subscription options	0	-
ws69	vertical payment service4	vp subscription options	price for subscription	0	-

ws70	vertical payment service5	VP id	demographic filters,autoresponse options,email aliases options	0	-
ws71	vertical payment service6	test drive id	duration alloted	0	-
ws72	product campaigning service1	pcs user id/pswd	ssr id	0	-
ws73	product campaigning service2	ssr id	campaigning options	0	-
ws74	product campaigning service3	product for campaigning	list of supporting links	0	-
ws75	product campaigning service4	ssr id	spam test option	0	-
ws76	product campaigning service5	ssr id	demographic reports	0	-
ws77	mail crimp service1	mail crimp user name/pswd	template wizards	0	-
ws78	mail crimp service2	template wizards	template online purchase options/price	0	-
ws79	mail crimp service3	mail crimp user name/pswd	set up wizard	0	-
ws80	mail crimp service4	set up wizard	set up options	0	-
ws81	mail crimp service5	mail crimp user name/pswd	campaign wizard	0	-
ws82	Mail crimp service5	Campaign wizard	Online campaign cost	0	-
ws83	Mail crimp service7	Mail crimp user name/pswd	Mc coupon id	0	-
ws84	Mail crimp service10	Coupon id	feature	0	-
ws85	mail crimp service9	feature	Test use validity period/download limit	0	-
ws86	Email brain service1	EB user name/pswd	coupon id	0	-
ws87	Email brain service2	coupon id	features	0	-
ws88	Email brain service3	feature	Test use validity period/download limit	0	-
ws89	Email brain service4	feature	list of existing customer contacts	0	-
ws90	Email brain service5	feature	available options	0	-
ws91	Email brain service6	coupon id	interface options	0	-
ws92	Email brain service7	coupon id	newsletter options	0	-
ws93	boomerang service1	boomerang id	questionnaire option	0	-

ws94	boomerang service2	questionnaire id	number of customers attempted questionnaire	0	-
ws95	boomerang service3	boomerang id	feedback	0	-
ws96	boomerang service5	boomerang id	FAQs	0	-
ws97	boomerang service6	FAQs	answers	0	-
ws98	boomerang service7	feedback	popular services/most unliked services	0	-
ws99	boomerang service8	boomerang id	coupon id	0	-
ws100	boomerang service9	feature	Test use validity period/download limit	0	-
ws101	Constant contact service3	Business domain	Newsletter template	0	-

Table 4.1 List of E-Marketing services

Set I

SNO	INPUT CATEGORY	OUTPUT CATEGORY
1	Request(work to be done)	Works done by company
2	Logo designing	Range of cost for logos
3	Tool for web designing	URL of the tool
4	Product name	User review
5	Country code, principal amount	Amount after tax reduction
6	Icontact id	price
7	Pinpointer id	Answers for FAQs
8	Mail crimp user name/pswd	Set up options
9	Boomerang id	feedback
10	EB User name/ pswd	Test use validity period/ download limit

Table 4.2 - User queries-Set I

Process model generated (after service composition) for set 1 queries

SNO	INPUT CATEGORY	OUTPUT CATEGORY	PROCESS MODEL GENERATED
1	Request(work to be done)	Works done by company	WS03, WS04, WS05
2	Logo designing	Range of cost for logos	WS06, WS09
3	Tool for web designing	URL of the tool	WS02 (single service)
4	Product name	User review	WS19, WS24
5	Country code, principal amount	Amount after tax reduction	WS28 (single service)
6	Icontact id	price	WS51, WS52, WS53
7	Pinpointer id	Answers for FAQs	WS62, WS63
8	Mail crimp user name/pswd	Set up options	WS79, WS80
9	Boomerang id	feedback	WS95 (single service)
10	EB User name/ pswd	Test use validity period/ download limit	WS86, WS87, WS88

Table 4.3 - Process model generated (after service composition) for set 1 queries

Success rate calculation for set I queries

S.No	With composition:	Without composition:
1	No.of services=20 Success rate = no. of answerablequeries/10 =2/10 =0.2	No. of services = 20 Success rate = 2/10 = 0.2
2	No.of services=40 Success rate = 3/10 = 0.3	No. of services = 20 Success rate = 2/10 = 0.2
3	No.of services=60 Success rate = 4/10 = 0.4	No. of services = 20 Success rate = 2/10 = 0.2
4	No.of services=80 Success rate = 6/10 = 0.6	No. of services = 20 Success rate = 2/10 = 0.2
5	No.of services = 100 Success rate = 7/10 = 0.7	No. of services = 20 Success rate = 2/10 = 0.2

Table 4.4

Performance graph

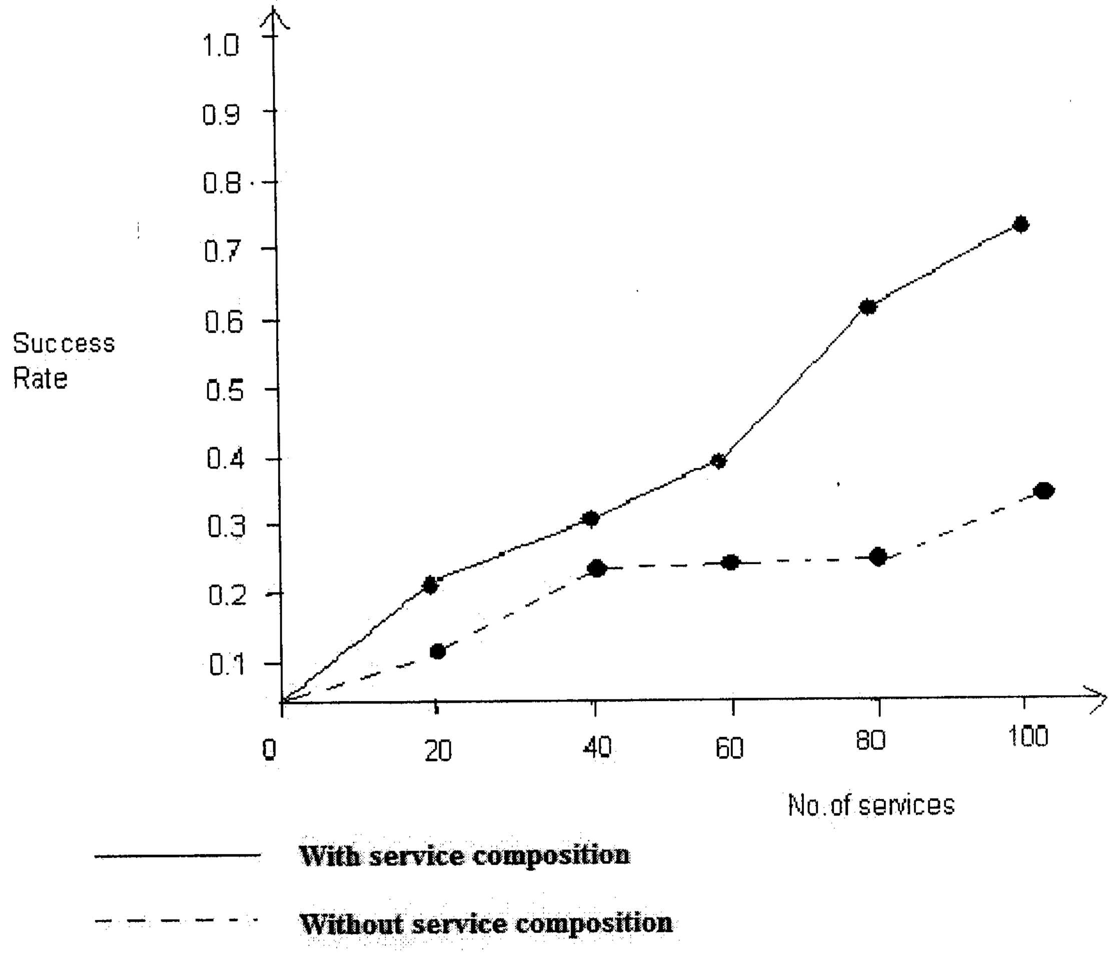


Figure 4.1

Set II

SNO	INPUT CATEGORY	OUTPUT CATEGORY
1	Web based solution(purpose)	URL OF THE TOOL
2	Transportation type	Estimate Cost
3	City zip code	Show time
4	Business type, business domain	Cost of template
5	Country code	Foreign exchange rate
6	City name	Location of jewel shop
7	Pcs User id/ pswd	Campaigning options
8	Test drive id	Duration allotted
9	EB User name/ pswd	Newsletter options
10	Boomerang id	Popular services/most unliked services

Table 4.5 User Queries-Set II

Process model generated (after service composition) for Set II queries

SNO	INPUT CATEGORY	OUTPUT CATEGORY	PROCESS MODEL GENERATED
1	Web based solution(purpose)	URL OF THE TOOL	WS01,WS02
2	Transportation type	Estimate Cost	WS16,WS17
3	City zip code	Show time	NO SERVICE
4	Business type, business domain	Cost of template	WS33,WS34 || WS101, WS34 (Parallel model)
5	Country code	Foreign exchange rate	WS39 (single service)
6	City name	Location of jewel shop	WS47,WS48,WS50
7	Pcs User id/ pswd	Campaigning options	WS72, WS73
8	Test drive id	Duration allotted	WS71 (single service)
9	EB User name/ pswd	Newsletter options	WS86, WS87, WS92
10	Boomerang id	Popular services/most unliked services	WS95, WS98

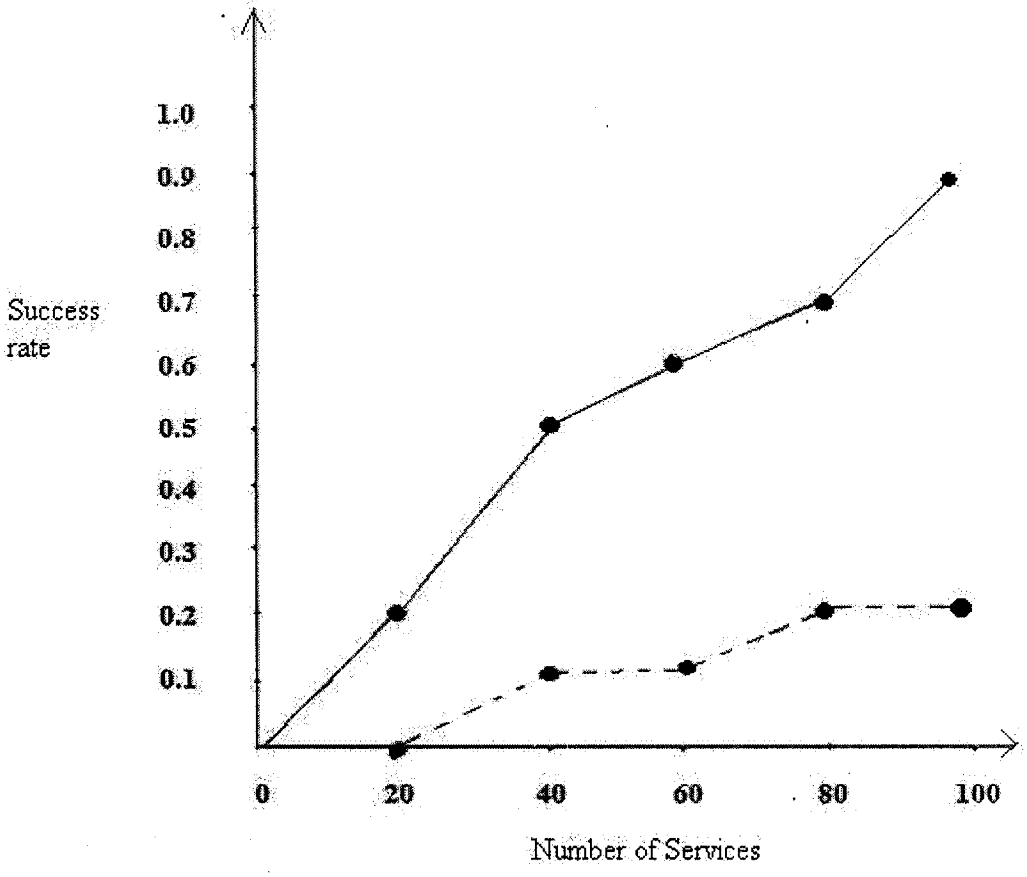
Table 4.6 - Process model generated (after service composition) for set II queries

Success rate calculation for set II queries

S.No	With composition:	Without composition:
1	Set 1: No.of services=20 Success rate = no. of answerablequeries/10 =2/10 =0.2	Set 1: No. of services = 20 Success rate = 0/10 = 0
2	No.of services=40 Success rate = 5/10 = 0.5	No. of services = 20 Success rate = 1/10 = 0.1
3	No.of services=60 Success rate = 6/10 = 0.6	No. of services = 20 Success rate = 1/10 = 0.1
4	No.of services=80 Success rate = 7/10 = 0.7	No. of services = 20 Success rate = 2/10 = 0.2
5	No.of services = 100 Success rate = 9/10 = 0.9	No. of services = 20 Success rate = 2/10 = 0.2

Table 4.7

Performance Graph



——— **With service composition**

– – – – **Without service composition**

Figure 4.2

CHAPTER V

5. CONCLUSION AND FUTURE WORK

5.1 Conclusion

To facilitate wider use of sensor data and to motivate more innovative applications, we proposed an efficient algorithm which composes dynamically, 'n' number of services available in the sensor nodes to satisfy the user's query. In order to increase the efficiency of the sensor nodes, we have proposed an algorithm to minimize the response time required to process the query.

Thus through the experiments, we show that the algorithm is efficient in terms of shorter response time, and enhances the performance of the sensor nodes.

5.2 Future work

Service composition in wireless sensor networks has a number of issues to be dealt witl There are many issues related to service composition especially when it is implemented in wireles sensor network. Dealing with request calls in a more sophisticated way requires advanced plannin which in turn requires more advanced approaches to performance optimization. Only littl functionalities have been implemented in our project. We set our next goal towards implementin innovative solutions to focus on Qos parameters in service composition, in order to realize mor benefits.

APPENDIX

SOURCE CODE

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.OracleClient;
using System.Data.Odbc;
using System.Data.OleDb;
using Microsoft.VisualBasic;
using System.Diagnostics;

namespace WindowsFormsApplication2
{

public partial class Form1 : Form
{
int pt1 = 600;
int pt2 = 76;
int pt3 = 600;
int pt4 = 159;
int i = 0;
int i1 = 0;
string str2 = "";
string str1;
string str3;
string str4 = "";
ComboBox[] cb = new ComboBox[10];
ComboBox[] cb1 = new ComboBox[10];
int z = 0;
int y = 0;
string[] temp = new string[7];
int ptr1 = 600;
int ptr2 = 50;
int ptr3 = 200;
int ptr4 = 50;
int fpt1 = 655;
int fpt2 = 55;
int h, k = 0;
public Form1()
{

InitializeComponent();


}
public Form1(int kkkk)
{
   int g = kkkk;
   InitializeComponent();
   trial(g);


}
public void trial(int g)
{
   if (g == 0)
   {
      comboBox1.Items.Add("pinpointer user id");
```

```
comboBox1.Items.Add("subscription options");
comboBox1.Items.Add("pinpointer id");
comboBox1.Items.Add("bulk mail options");
comboBox1.Items.Add("FAQs");
comboBox1.Items.Add("online training sections");
comboBox1.Items.Add("mail crimp user name/pswd");
comboBox1.Items.Add("template wizards");
comboBox1.Items.Add("set up wizard");
comboBox1.Items.Add("campaign wizard");
comboBox1.Items.Add("coupon id");
comboBox1.Items.Add("feature");
comboBox1.Items.Add("EB user name/pswd");
comboBox1.Items.Add("boomerang id");
comboBox1.Items.Add("questionnaire id");
comboBox1.Items.Add("feedback");
comboBox1.Items.Add("blog id");
comboBox1.Items.Add("online survey");

comboBox2.Items.Add("subscription options");
comboBox2.Items.Add("price");
comboBox2.Items.Add("bulk mail options");
comboBox2.Items.Add("existing customer email ids");
comboBox2.Items.Add("FAQs");
comboBox2.Items.Add("answers");
comboBox2.Items.Add("online training sections");
comboBox2.Items.Add("training duration");
comboBox2.Items.Add("template wizards");
comboBox2.Items.Add("template online purcahse/price");
comboBox2.Items.Add("set up wizard");
comboBox2.Items.Add("set up options");
comboBox2.Items.Add("campaign wizard");
```

```
    comboBox2.Items.Add("online campaign cost");
    comboBox2.Items.Add("Mc coupon id");
    comboBox2.Items.Add("feature");
    comboBox2.Items.Add("test use validity period/download limit");
    comboBox2.Items.Add("coupon id");
    comboBox2.Items.Add("list of existing customer contacts");
    comboBox2.Items.Add("available options");
    comboBox2.Items.Add("interface options");
    comboBox2.Items.Add("newsletter options");
    comboBox2.Items.Add("questionnaire option");
    comboBox2.Items.Add("number of customers attempted questionnaire");
    comboBox2.Items.Add("feedback");
    comboBox2.Items.Add("popular services/most unliked services");
    comboBox2.Items.Add("online survey");
    comboBox2.Items.Add("rating among blogs");
    comboBox2.Items.Add("number of followers");
}
else if (g == 1)
{
    comboBox1.Items.Add("transportation type");
    comboBox1.Items.Add("load");
    comboBox1.Items.Add("route");
    comboBox1.Items.Add("product name");
    comboBox1.Items.Add("product id");
    comboBox1.Items.Add("country name");
    comboBox1.Items.Add("country code");

    comboBox2.Items.Add("load");
    comboBox2.Items.Add("estimate cost");
    comboBox2.Items.Add("transportation type");
    comboBox2.Items.Add("product id");
    comboBox2.Items.Add("use(purpose)");
    comboBox2.Items.Add("rating");
    comboBox2.Items.Add("cost");
    comboBox2.Items.Add("URL for download");
    comboBox2.Items.Add("user review");
    comboBox2.Items.Add("cost of travel");
    comboBox2.Items.Add("country code");
    comboBox2.Items.Add("sales tax percentage for different range of amounts");
    comboBox2.Items.Add("amount after tax reduction");
}
else if (g == 2)
{
    comboBox1.Items.Add("city name");
    comboBox1.Items.Add("city zip");
    comboBox1.Items.Add("country name");
    comboBox1.Items.Add("country code");
    comboBox1.Items.Add("card number");

    comboBox2.Items.Add("city zip");
    comboBox2.Items.Add("latitude");
    comboBox2.Items.Add("location");
    comboBox2.Items.Add("country code");
    comboBox2.Items.Add("rate of metals available");
    comboBox2.Items.Add("card type");
    comboBox2.Items.Add("validity");

}
else if (g == 3)
{
    comboBox1.Items.Add("request (work to be done)");
    comboBox1.Items.Add("company name providing the solution");
```

```
comboBox1.Items.Add("company name");
comboBox1.Items.Add("business type");
comboBox1.Items.Add("logo designing options");
comboBox1.Items.Add("business card designing options");
comboBox1.Items.Add("letter head designing options");
comboBox1.Items.Add("logo templates");
comboBox1.Items.Add("business card templates");
comboBox1.Items.Add("letter templates");
comboBox1.Items.Add("envelope templates");
comboBox1.Items.Add("envelope designing options");
comboBox1.Items.Add("brochure designing options");
comboBox1.Items.Add("brochure templates");
comboBox1.Items.Add("icontact id");
comboBox1.Items.Add("coupon no");
comboBox1.Items.Add("features");
comboBox1.Items.Add("existing customers");
comboBox1.Items.Add("icontact id");
comboBox1.Items.Add("campaign options");


comboBox2.Items.Add("list of companies providing the solution");
comboBox2.Items.Add("company profile");
comboBox2.Items.Add("works done");
comboBox2.Items.Add("company profile");
comboBox2.Items.Add("logo templates");
comboBox2.Items.Add("business card templates");
comboBox2.Items.Add("letter templates");
comboBox2.Items.Add("envelope templates");
comboBox2.Items.Add("brochure templates");
comboBox2.Items.Add("range of cost");
comboBox2.Items.Add("newsletter template");
```

```
    comboBox2.Items.Add("coupon no");
    comboBox2.Items.Add("features");
    comboBox2.Items.Add("price for the feature");
    comboBox2.Items.Add("existing customers");
    comboBox2.Items.Add("renewal options");
    comboBox2.Items.Add("campaign options");
    comboBox2.Items.Add("cost of the product");
}
else if (g == 4)
{
    comboBox1.Items.Add("web based solution(purpose)");
    comboBox1.Items.Add("name of the tool");
    comboBox1.Items.Add("city name");
    comboBox1.Items.Add("zip code");
    comboBox1.Items.Add("list of jeweller shops");
    comboBox1.Items.Add("VP id");
    comboBox1.Items.Add("test drive id");
    comboBox1.Items.Add("VP subscription options");
    comboBox1.Items.Add("pcs user id/pswd");
    comboBox1.Items.Add("ssr id");
    comboBox1.Items.Add("product for campaigning");

    comboBox2.Items.Add("name of the tool");
    comboBox2.Items.Add("URL of the tool");
    comboBox2.Items.Add("list of jeweller shops");
    comboBox2.Items.Add("zip code");
    comboBox2.Items.Add("owner name");
    comboBox2.Items.Add("location of the shop");
    comboBox2.Items.Add("test drive options");
    comboBox2.Items.Add("test drive features");
    comboBox2.Items.Add("VP subscription options");
```

```
        comboBox2.Items.Add("price for subscription");
        comboBox2.Items.Add("demographic response");
        comboBox2.Items.Add("duration alloted");
        comboBox2.Items.Add("ssr id");
        comboBox2.Items.Add("campaigning options");
        comboBox2.Items.Add("list of supporting links");
        comboBox2.Items.Add("spam test option");
        comboBox2.Items.Add("demographic reports");
    }
    else if (g == 5)
    {
        comboBox1.Items.Add("city name");
        comboBox1.Items.Add("theatre names");
        comboBox1.Items.Add("theatre name");
        comboBox1.Items.Add("current movies");

        comboBox2.Items.Add("current movies");
        comboBox2.Items.Add("theatre names");
        comboBox2.Items.Add("show time");
        comboBox2.Items.Add("availability of tickets");
    }

}

private void button1_Click(object sender, EventArgs e)
{
string com = comboBox1.Text;
string in1 = com + str2;
MessageBox.Show("You Entered the INPUT:" + in1);
string com1 = comboBox2.Text;
string out1 = com1 + str4;
```

```
MessageBox.Show("You Entered the OUTPUT:" + out1);
mytable mt = new mytable();
temp = mt.sample(in1, out1);

for (int t = 0; t < temp.Length; t++)
{

if (temp[t] != null)
k++;


}

if (k != 0)
{
   Form1 form1 = new Form1();
   Form3 form3 = new Form3(temp, k);
   form3.BackColor = Color.White;
   form3.WindowState = FormWindowState.Maximized;

   form3.Show();
}
}

public void timer1_Tick(object sender, EventArgs e)
{

DialogResult dlgResult = MessageBox.Show("Time out.. Do you want to continue?", "Continue?",
MessageBoxButtons.YesNo, MessageBoxIcon.Question);
if (dlgResult == DialogResult.Yes)
{
}

else if (dlgResult == DialogResult.No)
{
this.Close();
}
}
private void button2_Click(object sender, EventArgs e)
{
addnew();
z++;
}

public void addnew()
{
cb[z] =new ComboBox();
cb[z].Items.Add("distance");
cb[z].Items.Add("principal amount");
cb[z].Items.Add("zip code");
cb[z].Items.Add("show time");
cb[z].Location =new System.Drawing.Point(pt1, pt2);
cb[z].Size =new System.Drawing.Size(140, 28);
cb[z].Font =new System.Drawing.Font("Century Schoolbook", 12F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
cb[z].SelectedValueChanged +=new EventHandler(OnTextChanged);
this.Controls.Add(cb[z]);
pt1 = pt1 + 150;
}


public void addnew1()
```

```
{
cb1[y] =new ComboBox();
cb1[y].Items.Add("training cost");
cb1[y].Items.Add("longitudinal location");
cb1[y].Items.Add("autoresponse options");
cb[y].Items.Add("email aliases options");
cb1[y].Location =new System.Drawing.Point(pt3, pt4);
cb1[y].Size =new System.Drawing.Size(140, 28);
cb1[y].Font =new System.Drawing.Font("Century Schoolbook", 12F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
cb1[y].SelectedValueChanged +=new EventHandler(OnTextChanged1);
this.Controls.Add(cb1[y]);
pt3 = pt3 + 150;
}

public void OnTextChanged(object sender, EventArgs e)
{
str1 ="," + cb[i].Text;
str2 += str1;
i++;
}

private void button3_Click(object sender, EventArgs e)
{
addnew1();
y++;
}
public void OnTextChanged1(object sender, EventArgs e)
{
str3 ="," + cb1[i1].Text;
str4 += str3;
```

```
i1++;
}
}

public class mytable : Form1
{
string id;
string name;
string inp;
string outp;
int tag;
string dep1;
List<string> dep = new List<string>();
List<string> depfwd = new List<string>();
public string[] sample(string req_in, string req_out)
{
int flag = 0;
int temp = 0;
int c = 0;
int t = 0;
string ws_id;
int i, k1,k2;
int m, w, nows;
Label[] lab = new Label[10];
Stopwatch timer = new Stopwatch();
int[] l = new int[5];
string[] tab = new string[7];
string[] tab1 = new string[7];
/*****************CREATION OF  SSI  TABLE*******************/
OdbcConnection conn = new OdbcConnection();
conn.ConnectionString ="Dsn=sample";
conn.Open();
OdbcCommand DbCommand = conn.CreateCommand();
OdbcCommand DbCommand1 = conn.CreateCommand();
OdbcCommand DbCommand2 = conn.CreateCommand();
string connString =
"Provider=Microsoft.Jet.OLEDB.4.0;Data Source=D://Data Store//Sandhya//final year//project-
WSN//sfrefr//db1.mdb;Persist Security Info=False";
string query = "SELECT * FROM services";
OleDbDataAdapter dAdapter = new OleDbDataAdapter(query, connString);
OleDbCommandBuilder cBuilder = new OleDbCommandBuilder(dAdapter);
DataTable dTable = new DataTable();
dAdapter.Fill(dTable);

nows = dTable.Rows.Count;
 mytable[] obje = new mytable[nows + 1];
for (i = 1; i <= nows; i++)
{
obje[i] =new mytable();
}
m = 1;
foreach (DataRow row in dTable.Rows)
{

foreach (var item in row.ItemArray)
{
obje[m].id =Convert.ToString(row.ItemArray[0]);
obje[m].name =Convert.ToString(row.ItemArray[1]);
obje[m].inp =Convert.ToString(row.ItemArray[2]);
obje[m].outp =Convert.ToString(row.ItemArray[3]);
obje[m].tag =Convert.ToInt32(row.ItemArray[4]);
obje[m].dep1 =Convert.ToString(row.ItemArray[5]);
```

```
}
m++;
}
timer.Start();
/********* BACKWARD DEPENDENCY LIST COMPUTATION**********/
for (i = 1; i <= nows; i++)
{
   if (obje[i].tag == 0)
   {
   for (int j = i - 1; j > 0; j--)
   {
      if (obje[i].inp == obje[j].outp)
      obje[i].dep.Add(obje[j].id);

      if (obje[i].outp == obje[j].inp)
      obje[j].dep.Add(obje[i].id);
   }
   }
}
/********* FORWARD DEPENDENCY LIST COMPUTATION**********/

for (i = 1; i <= nows; i++)
{
   if (obje[i].tag == 0)
   {
      for (int j = i - 1; j > 0; j--)
      {
         if (obje[i].inp == obje[j].outp)
            obje[j].depfwd.Add(obje[i].id);

         if (obje[i].outp == obje[j].inp)
```

```
            obje[i].depfwd.Add(obje[j].id);
        }
    }
}
/**********TIMER FOR REQUEST PROCESSING***********/
timer2 = new System.Timers.Timer(10000); //10 second timeout
timer2.Enabled = true;
timer2.AutoReset = false;
timer2.Elapsed += new ElapsedEventHandler(timer_Elapsed);



 /*********PROCESS MODEL GENERATION****************/
 /***************For already composed services***************/
 for (i = nows; i > 0; i--)
 {
 if (obje[i].tag == 1 && req_out == obje[i].outp && obje[i].inp == req_in)
 {
 MessageBox.Show(string.Format("THE PROCESS MODEL IS : \n {0}", obje[i].dep1));
 string[] p2 = obje[i].dep1.Split(',');
 return p2;
 }
 }
 /***************Single Services satisfying request*****************/
 for (i = 1; i <= nows; i++)
 {
    if (req_out == obje[i].outp)
    {
       if (obje[i].inp == req_in)
       {
        MessageBox.Show(String.Format(" ^^^^^^^^^^^^^^A single service satisfies the user
request^^^^^^^^^^^^^ \n {0} has to be invoked", obje[i].id));
        return tab;
      }
   }
}

/*****************Composition of service*********************/
for(i=1;i<=nows;i++)
{
   if (req_out == obje[i].outp)
   {
      int dep_length = obje[i].dep.Count;
      for (int k = 0; k < dep_length; k++)
      {
         flag = 0;
      ws_id = obje[i].dep[k];
      l1:    for (w = 1; w <= nows; w++)
         {
            if (ws_id == obje[w].id)
            {
               if (req_in == obje[w].inp)
               {
                  tab[t] = obje[w].id;
                  if (flag >= 1)
                  {
                     for (c = flag - 1; c >= 0; c--)
                     {
                        temp = l[c];
                        tab[++t] = obje[temp].id;
                     }
```

```
                }
                tab[++t] = obje[i].id;
                goto l2;
            }
            else
            {
                int dep_length1 = obje[w].dep.Count;
                for (int z = 0; z < dep_length1; z++)
                {
                    ws_id = obje[w].dep[z];
                    flag++;
                    l[c] = w;
                    c++;
                    goto l1;
                }
            }
        }

      }
    }
}
else
{
    if (req_in == obje[i].inp)
    {
        int len_fwd = obje[i].depfwd.Count;
        for (int g = 0; g < len_fwd; g++)
        {
            string ws_id1 = obje[i].depfwd[g];
            if (tab[t] == ws_id1)
            {
                tab[t++] = obje[i].id;
                goto l2;
             }
          }
       }
    }
}
timer.Stop();
string s = req_in;
string[] words = s.Split(',');
if (words[0].Length!=s.Length)
{
    for (k1 = 1; k1 <= nows; k1++)
    {
        if (words[0] == obje[k1].inp)
            if (req_out == obje[k1].outp)
            {


               goto o1;
            }


    }
o1: for (k2 = 1; k2 <= nows; k2++)
    {
        if (words[1] == obje[k2].inp)
            if (req_out == obje[k2].outp)
            {
               MessageBox.Show(string.Format(" THE PARALLEL PROCESS MODEL IS \n {0} ||
{1}", obje[k2].dep1, obje[k1].dep1));
               string strp = obje[k2].dep1 + ",/," + obje[k1].dep1;
                string[] p1 = strp.Split(',');
            return p1;
          }
      }
}

MessageBox.Show("!---------------------------No Service available---------------------------!");
return tab;
l2: MessageBox.Show(string.Format("THE PROCESS MODEL IS: \n {0} {1} {2} {3} {4}",
tab[0], tab[1], tab[2], tab[3], tab[4]));
MessageBox.Show(Convert.ToString( timer.Elapsed));

/*****************Entering new composite service into table***********/

DbCommand1.CommandText = "SELECT * FROM services where input='" + req_in + "' and
output='" + req_out + "'";
OdbcDataReader DbReader1 = DbCommand1.ExecuteReader();
if (DbReader1.Read() == false)
{
   int nor = Convert.ToInt32(dTable.Rows.Count);
   int nari = nor + 1;
   string wsid2 = "ws" + nari;
   MessageBox.Show(string.Format("the id is {0}", wsid2));
   MessageBox.Show(string.Format("the name is {0}", wsname));
   DbCommand.CommandText = "INSERT INTO   services(id,name,input,output,tag,dep)
 VALUES('" + wsid2 + "','" + wsname + "','" + req_in + "','" + req_out + "','1','')";
   OdbcDataReader DbReader = DbCommand.ExecuteReader();
   DbReader.Close();
   int co = 0;
   for (i = 0; i < tab.Length; i++)
   {
      if (tab[i] != null)
```

```
        co++;
    }
      for (i = 0; i < co-1; i++)
    {
       tab[i] = tab[i] + ",";
    }
        for (i = 0; i < co; i++)
        {
            DbCommand2.CommandText = "UPDATE services SET dep = CONCAT(dep,'" + tab[i]
+ "' ) where id = '" + wsid2 + "' ";
        OdbcDataReader DbReader2 = DbCommand2.ExecuteReader();
        DbReader2.Close();
    }
    MessageBox.Show("A NEW COMPOSITE SERVICE HAS BEEN ADDED TO THE
TABLE");
}
DbCommand.Dispose();
DbCommand2.Dispose();
conn.Close();
return tab;
}
}
}
```

FLOW DIAGRAM GENERATION

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace WindowsFormsApplication2
{
   public partial class Form3 : Form
   {
      public Form1 form1;
      int pt1, ppt1;
      int pt2, ppt2;
      int pt3, ppt3;
      int pt4, ppt4;
      int fpt1, pfpt1;
      int fpt2, pfpt2;
      int lpt1, lpt2, plpt1, plpt2;
      int i, j, h, par = 0, l = 0,m=0,f;
      int count = 0;
      string[] temp1 = new string[10];
      string[] temp2 = new string[10];
      public Form3(string[] tem, int k)
      {
         InitializeComponent();
         l = 0;
         h = 0;
         for ( f = 0; f < k; f++)
         {
            if (tem[f] == "/")
            {
               for (j = f + 1; j < k; j++)
               {
                  this.temp2[h++] = tem[j];
            count++;
          }
          f = k;
          par = 1;
        }
      else
      {
          this.temp1[f] = tem[f];
          l++;
      }
    }
if (par == 1)
    {
      if (temp1[l - 1]== temp2[count - 1] )
      {
          m = 1;
          l--;
      }
    }
 }

 protected override void OnPaint(PaintEventArgs e)
 {
    /***********SEQUENTIAL MODEL DECLARATIONS************/
    pt1 = 250;
    pt2 = 150;
    pt3 = 150;
    pt4 = 50;
    fpt1 = 255;
    fpt2 = 155;
    lpt1 = 200;
```

```
  lpt2 = 250;
  /**********PARALLEL MODEL DECLARATIONS***********/
  ppt1 = 550;
  ppt2 = 150;
  ppt3 = 150;
  ppt4 = 50;
  pfpt1 = 555;
  pfpt2 = 155;
  plpt1 = 200;
  plpt2 = 250;
  draw1();
}
void draw1()
{
  System.Drawing.Graphics graphicsObj;
  graphicsObj = this.CreateGraphics();
  Pen myPen = new Pen(System.Drawing.Color.CadetBlue, 5);
  Pen myPen1 = new Pen(System.Drawing.Color.DarkOrchid, 5);
  Font myFont = new System.Drawing.Font("Helvetica", 28, FontStyle.Italic);
  Brush myBrush = new SolidBrush(System.Drawing.Color.DarkOliveGreen);
   /********GRAPH FOR SEQUENTIAL MODEL********/
  for (i = 0; i < l ; i++)
  {
    Rectangle myRectangle = new Rectangle(pt1, pt2, pt3, pt4);
    graphicsObj.DrawRectangle(myPen, myRectangle);
    graphicsObj.DrawString(temp1[i], myFont, myBrush, fpt1, fpt2);
     pt2 += 100;
     fpt2 += 100;
  }
 for (i = 0; i < l - 1; i++)
  {
```

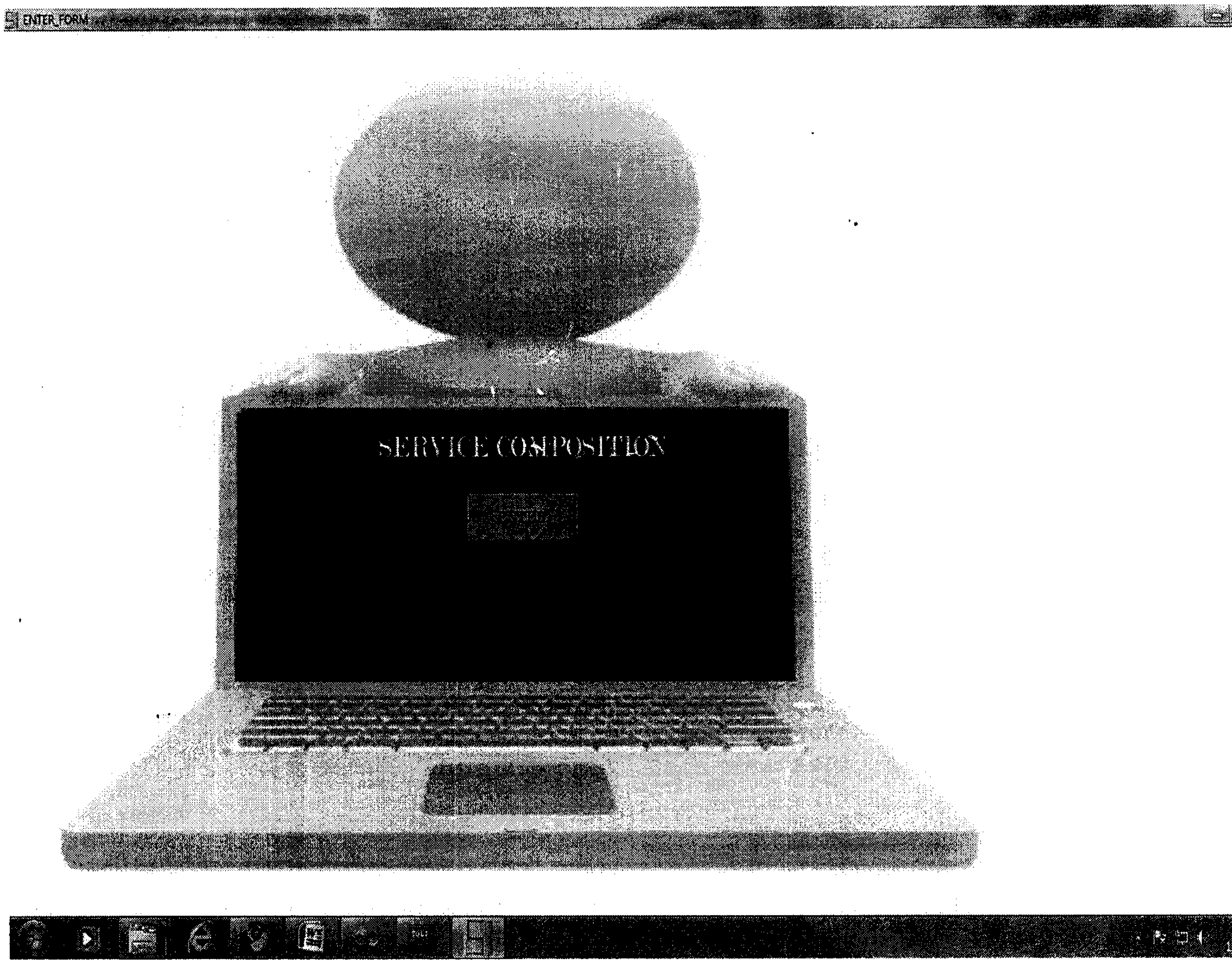
```
 graphicsObj.DrawLine(myPen1, 320, lpt1, 320, lpt2);
   lpt1 += 100;
   lpt2 += 100;
}

 /********GRAPH FOR PARALLEL MODEL************/
if (par == 1)
{
   for (i = 0; i < count ; i++)
   {
     if (temp1[1]== temp2[i] )
      {
        Rectangle myRectangle1 = new Rectangle(350, ppt2, 150, ppt4);
        graphicsObj.DrawRectangle(myPen, myRectangle1);
        graphicsObj.DrawString(temp2[i], myFont, myBrush, 355, pfpt2);
        goto l2;
      }
      Rectangle myRectangle = new Rectangle(ppt1, ppt2, ppt3, ppt4);
      graphicsObj.DrawRectangle(myPen, myRectangle);
      graphicsObj.DrawString(temp2[i], myFont, myBrush, pfpt1, pfpt2);
      ppt2 += 100;
      pfpt2 += 100;
   }
 l2:  for (i = m; i < count - 1; i++)
    {
      graphicsObj.DrawLine(myPen1, 620, plpt1, 620, plpt2);
      plpt1 += 100;
      plpt2 += 100;
    }
    graphicsObj.DrawLine(myPen1, 320, lpt1, 420, 350);
    graphicsObj.DrawLine(myPen1, 620, plpt1, 420, 350);
```

```
        }
        myPen.Dispose();
        myPen1.Dispose();
        myBrush.Dispose();
        graphicsObj.Dispose();
      }
    }
}
```

SNAPSHOTS

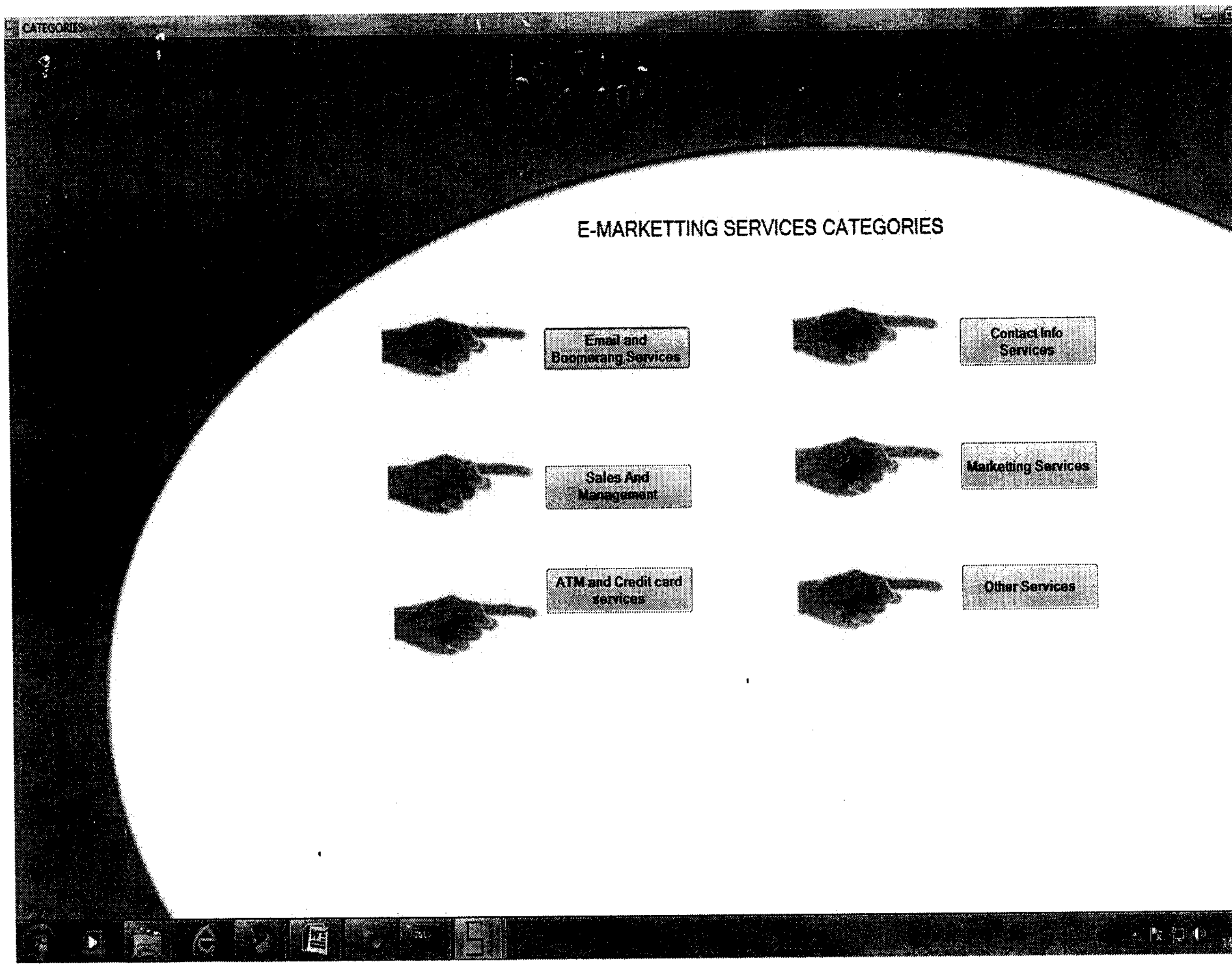
HOME PAGE



Enter form

Categories form

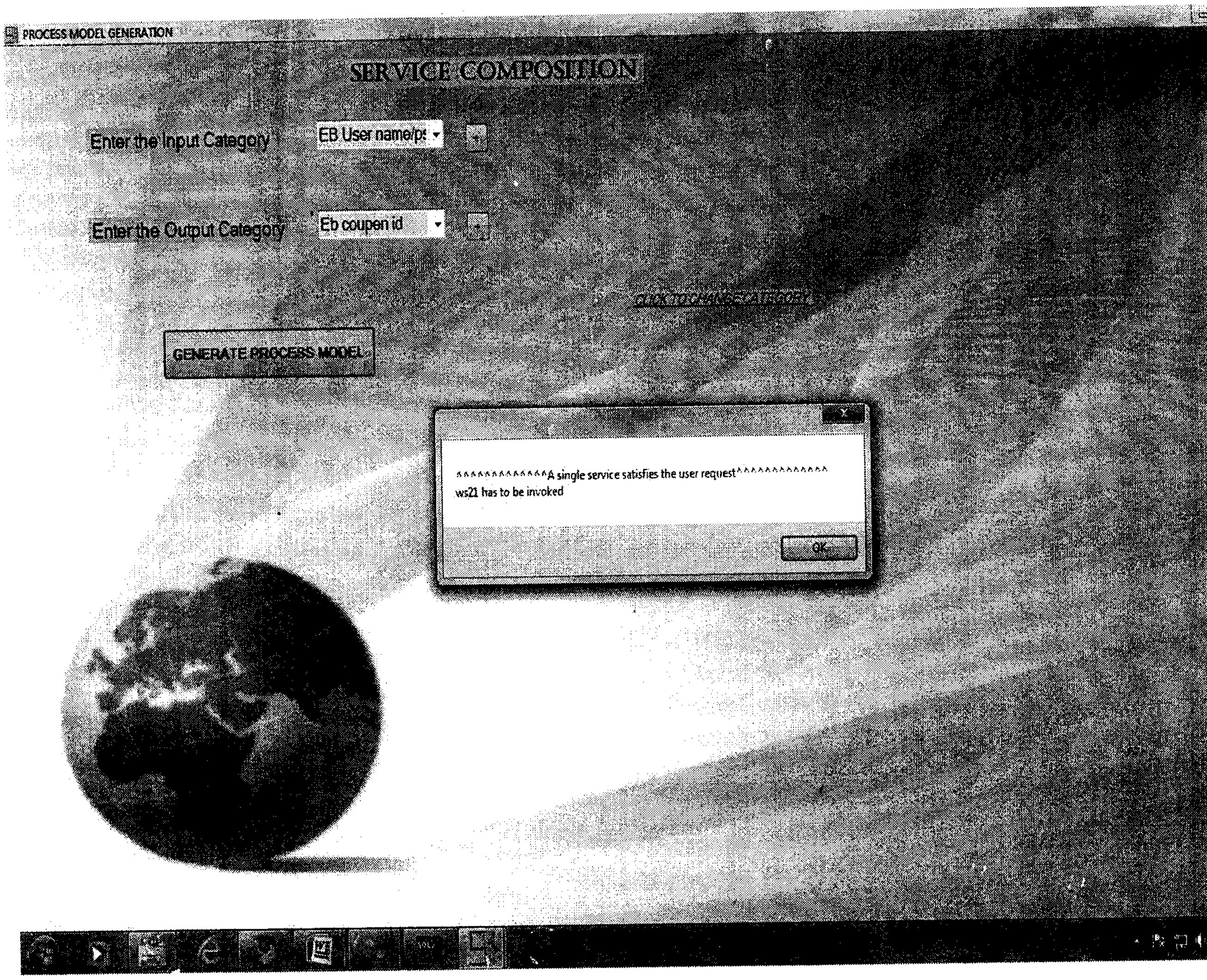
This allows the user to choose the category of services available in E-Marketing.


CATEGORIES
E-MARKETTING SERVICES CATEGORIES
Email and Boomerang Services
Contact Info Services
Sales And Management
Marketting Services
ATM and Credit card services
Other Services

PROCESS MODEL GENERATION

SEQUENTIAL MODEL

Case 1: A single service satisfies the user's request

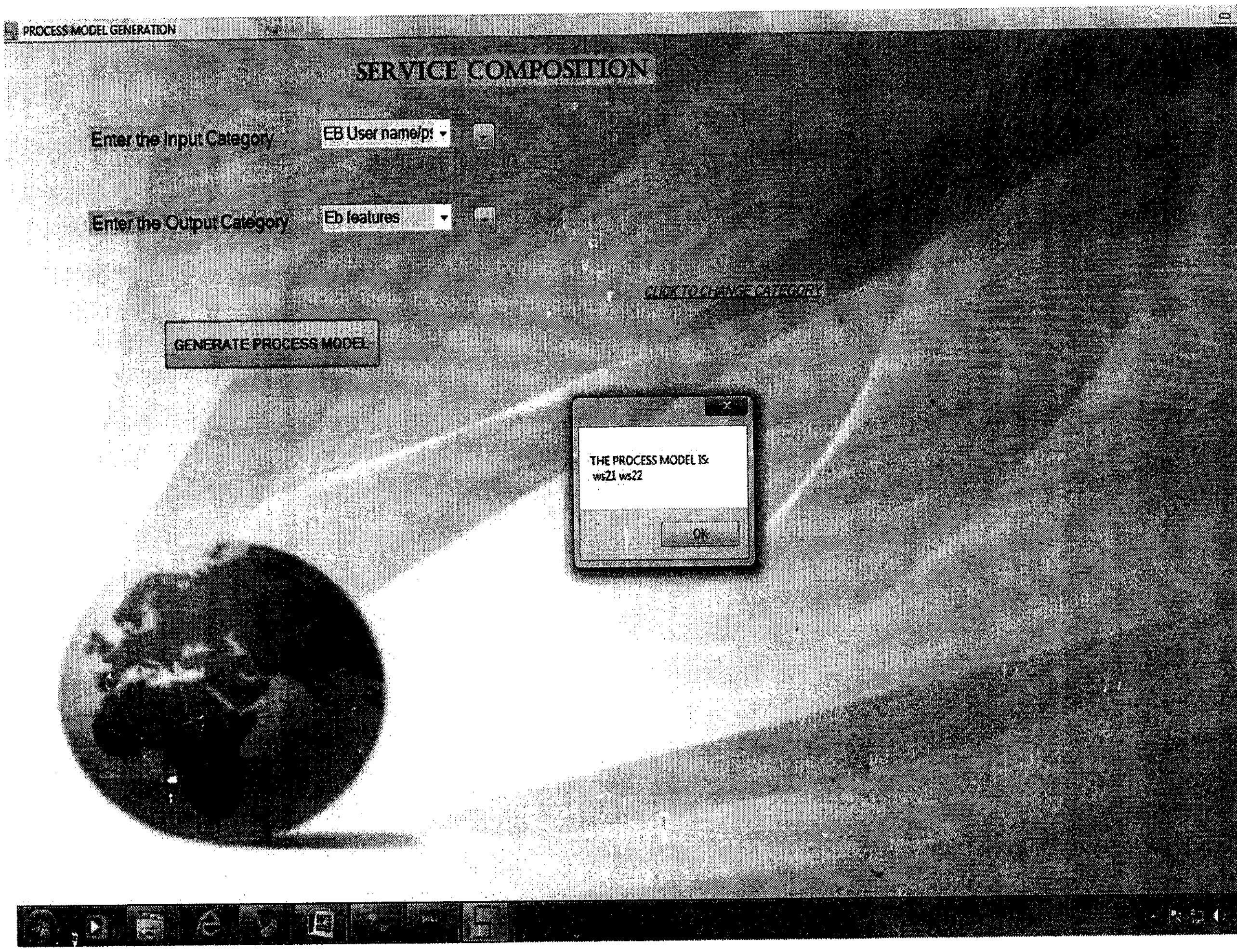


Enter the Input Category: EB User name/ps

Enter the Output Category: Eb coupen id

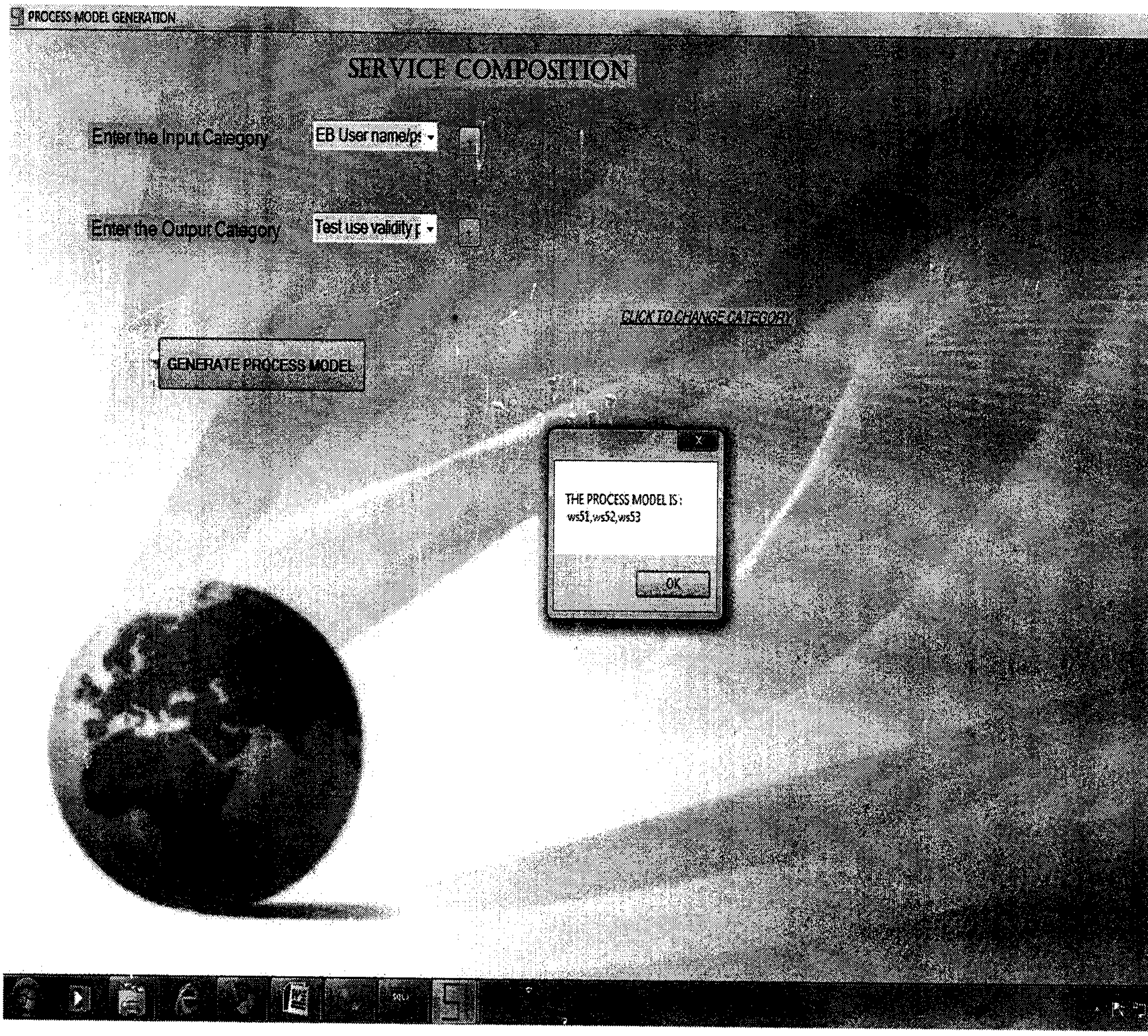
```
^^^^^^^^^^^^^A single service satisfies the user request^^^^^^^^^^^^^
ws21 has to be invoked
```

Case 2: Composition of 2 services in order to satisfy user's request



THE PROCESS MODEL IS:
ws21 ws22

Case 3: Composition of 3 services to satisfy the user's request

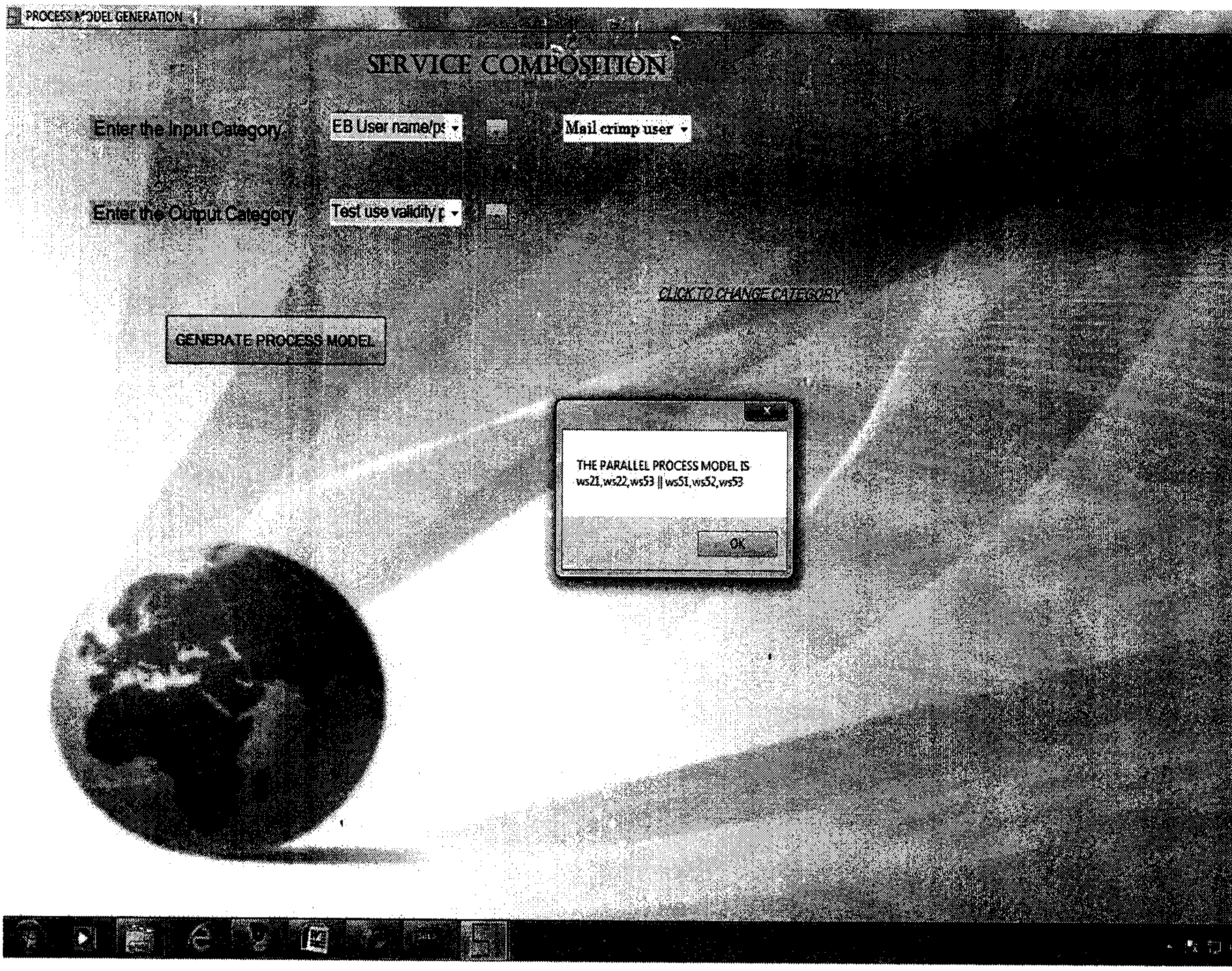


EB User name/ps

Test use validity p

THE PROCESS MODEL IS :
ws51,ws52,ws53

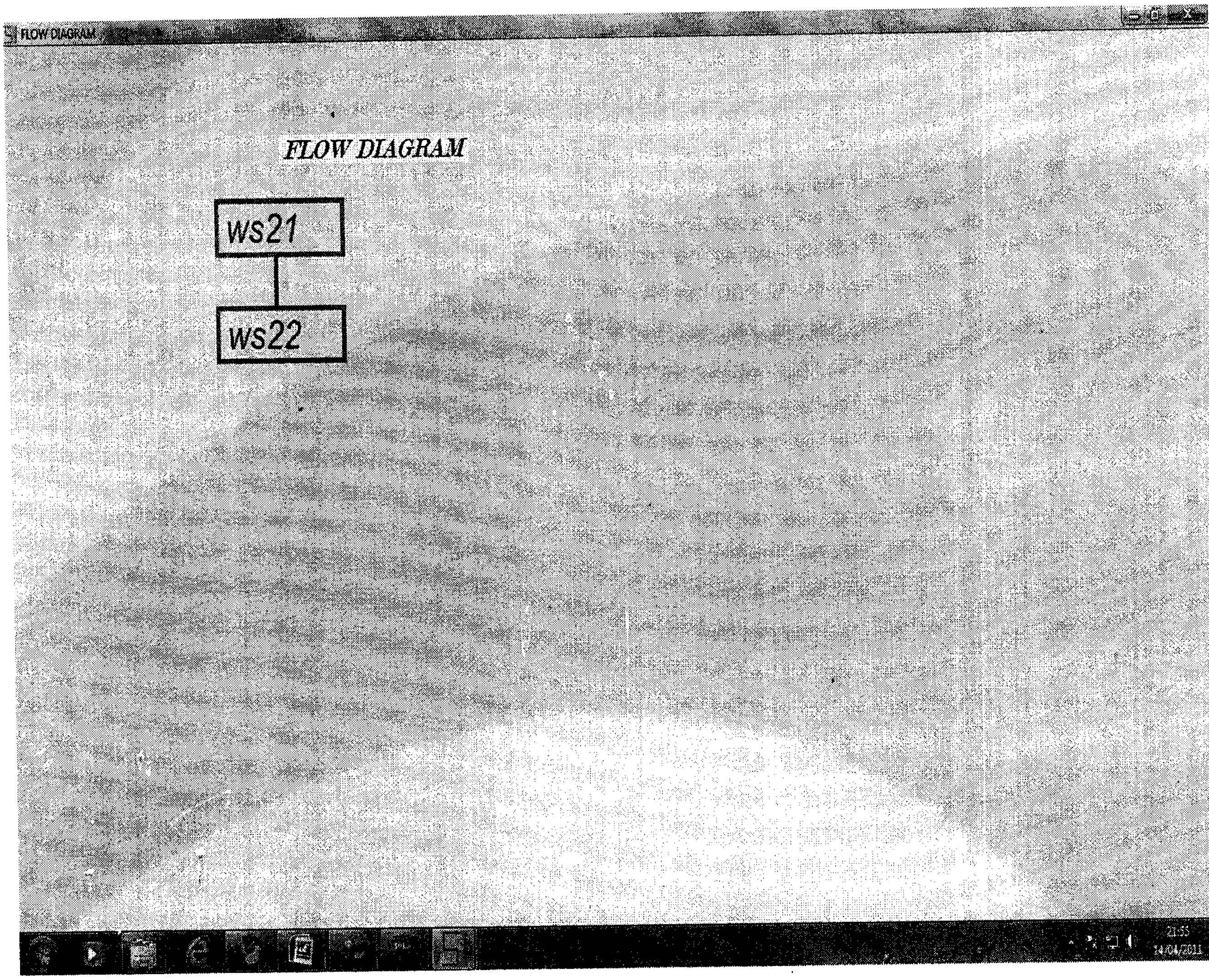
PARALLEL MODEL



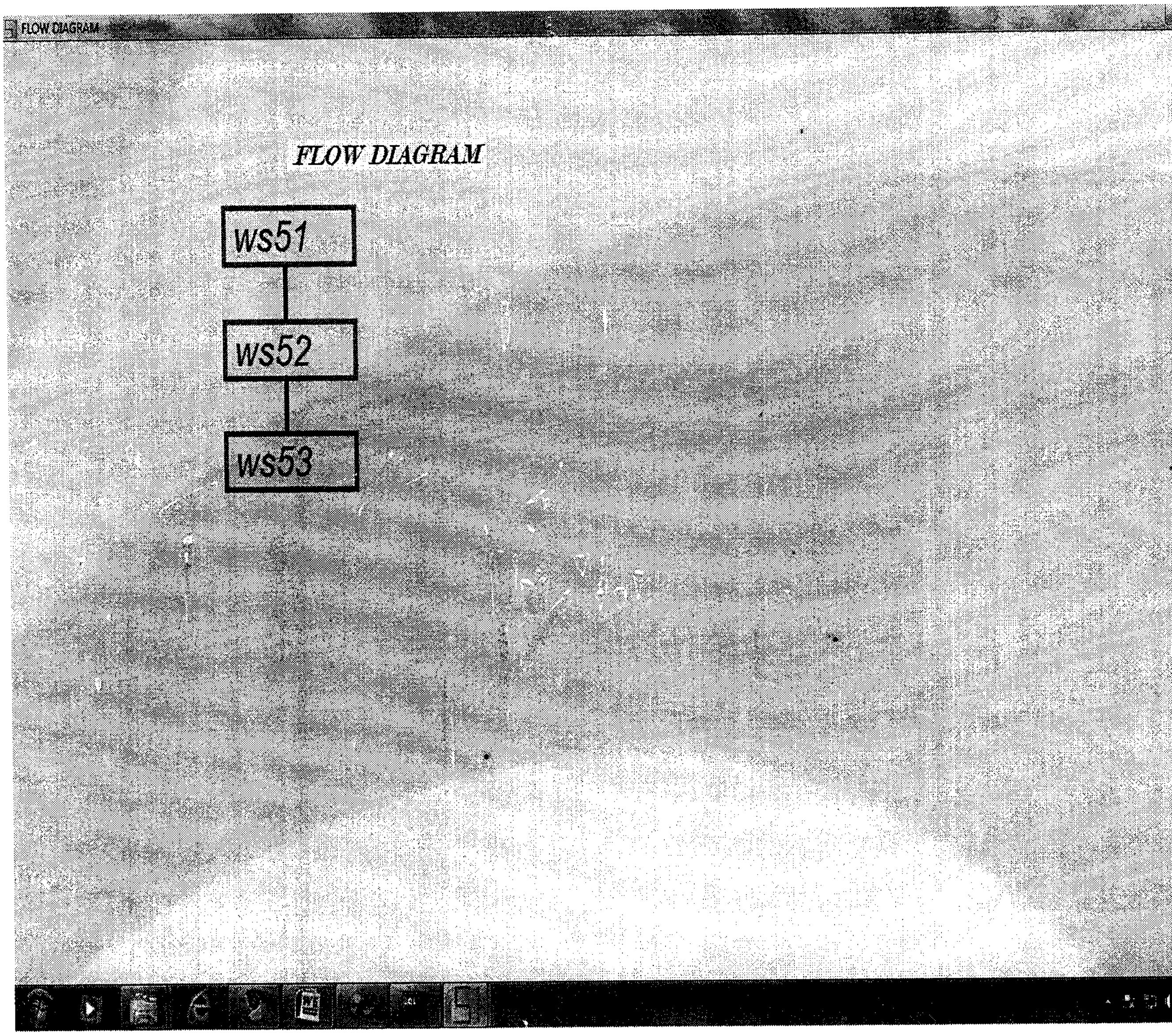
THE PARALLEL PROCESS MODEL IS
ws21,ws22,ws53 || ws51,ws52,ws53

FLOW DIAGRAMS

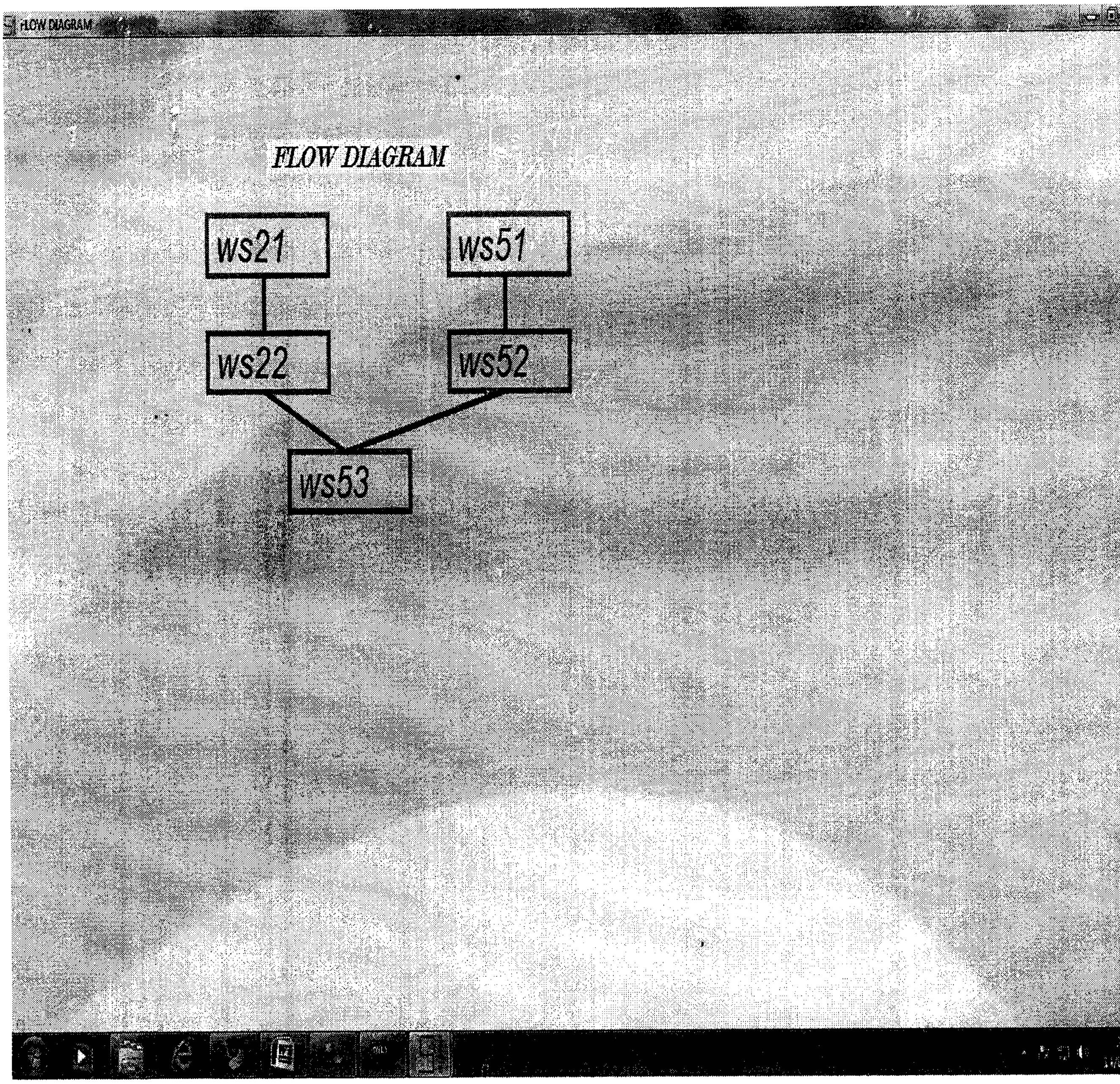
Composition of 2 services



Composition of 3 services



Flow diagram for parallel model



7. REFERENCES

[1] Abrehet Mohammed Omer, Alexander Schill, "Web services Composition using Input/Output Dependency Matrix", Proceedings of the 3rd workshop on Agent-oriented software engineering challenges for ubiquitous and pervasive computing (ACM AUPC) pp: 21-26 -2009.

[2] F. Akyildiz, W. Su, Y. Sankarsubramaniam, and E. Cayirci, "Wireless Sensor Networks: A Survey", Computer Networks, vol. 38, pp. 393-422, Mar. 2002.

[3] Leguay, J.; Lopez-Ramos, M.; Jean-Marie, K.; Conan, V," An efficient service oriented architecture for heterogeneous and dynamic wireless sensor networks", Local Computer Networks, 2008. LCN 2008. 33rd IEEE Conference , ISBN: 978-1-4244-2412-2.

[4] Lian Li, Ma Jun, Chen ZhuMin, Song Ling, "An Efficient Algorithm for Web service Composition with a Chain data structure", Proceedings of the IEEE Asia-Pacific conference on service computing (APSCC'06) – 2006.

[5] Liu AnFeng Chen ZhiGang He Hui Gui WeiHua Central South Univ. Changsha "Treenet:A Web Services Composition Model Based on Spanning tree" Pervasive Computing and Applications, 2007. ICPCA 2007. 2nd Internationa Conference on 26-27 July 2007 pp:618-623, 2007.

[6] Xiumin Wang, Jianping wang, Zeyu Zheng, Yinlong Xu, Mej Yang, "Servic composition in service oriented Wireless sensor networks with persistent queries" Proceedings of the 6th IEEE Conference on Consumer Communications an Networking Conference, pp.368-372 , 2009.

[7] Yue Zhang ," Service Oriented Management of Heterogeneous Wireless Sensor Networks", Wireless Communications, Networking and Mobile Computing, 2009. WiCom '09. 5th International Conference, ISBN: 978-1-4244-3692-7.

[8] Zhumin Chen, Jun Ma, Ling Song, Li Lian, "An Efficient Approach to web service Discovery and Composition when Large Scale Services are Available." , Proceedings of the IEEE Asia-Pacific conference on service computing(APSCC'06) – 2006.