Fuzzy Rule Based Approach for Quality Analysis of Web Service Composition using Complexity Metrics

Debendra Kumar Naik

Roll. 213CS3188

under the supervision of

Prof. Santanu Kumar Rath

Department of Computer Science and Engineering

National Institute of Technology Rourkela

Rourkela – 769008, India

Fuzzy Rule Based Approach for Quality Analysis of Web Service Composition using Complexity Metrics

Thesis submitted in partial fulfillment of the requirements for the degree of

Master of Technology

in

Computer Science and Engineering

(Specialization: Software Engineering)

by

Debendra Kumar Naik

(Roll No.- 213CS3188)

under the supervision of

Prof. S. K. Rath

Department of Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela, Odisha, 769 008, India

June 2015

Computer Science and Engineering

National Institute of Technology Rourkela

Rourkela-769 008, India. `www.nitrkl.ac.in`

MAY 23, 2015

Certificate

This is to certify that the work in the thesis entitled *Fuzzy Rule Based Approach for Quality Analysis of Web Service Composition using Complexity Metrics* by *Debendra Kumar Naik*, having roll number 213CS3188, is a record of an original research work carried out by him under my supervision and guidance in partial fulfillment of the requirements for the award of the degree of *Master of Technology* in *Computer Science and Engineering Department*. Neither this thesis nor any part of it has been submitted for any degree or academic award elsewhere.

Dr. Santanu Kumar Rath
Department of CSE
NIT Rourkela

Acknowledgment

First of all, I would like to express my deep sense of respect and gratitude towards my supervisor Prof. Santanu Kumar Rath, who has been the guiding force behind this work. I want to thank him for introducing me to the field of Service Oriented Architecture and giving me the opportunity to work under him. His undivided faith in this topic and ability to bring out the best of analytical and practical skills in people has been invaluable in tough periods. Without his invaluable advice and assistance it would not have been possible for me to complete this thesis. I am greatly indebted to him for his constant encouragement and invaluable advice in every aspect of my academic life. I consider it my good fortune to have got an opportunity to work with such a wonderful person.

I would also like to thank all faculty members, PhD scholars, my seniors and juniors and all colleagues to provide me their regular suggestions and encouragements during the whole work.

At last but not the least I am in debt to my family to support me regularly during my hard times.

I wish to thank all faculty members and secretarial staff of the CSE Department for their sympathetic cooperation.

Debendra Kumar Naik
213CS3188

Abstract

Since the human needs are fast changing, the present day software tends to be complex. So, complexity analysis of any software is the one of the challenging areas of research. In the literature review, a good number of articles are available on traditional software complexity analysis; but the complexity analysis of service oriented architecture based software is not studied extensively till date. The web service is the basic building block of SOA. Composition of web service is done through a Business Process Execution Language; but a large number of web service compositions make the software more complex. So, it is necessary to analyze the complexity of BPEL processes.

Business activities govern long-running complex composed service. That reduces the service reliability, performability, and others quality attributes. Business process complexity metrics are considered for analysis of composed web service. In this work different complexity metrics are proposed and Fuzzy logic is used for quality analysis of web service composition. This model relates business complexity metrics such as activity complexity, structural complexity, control flow complexity to high-level quality attributes such as functionality, usability, maintainability, reliability, performability using fuzzy rule based approach.

Keywords: BPEL process, web service, complexity, performance, Fuzzy logic, Quality model.

Contents

List of Figures

List of Tables

List Of Abbreviation

SOA	Service oriented architecture
BPEL	Business process execution language
BPMN	Business process modelling language
BPCL	Business process choreography language
FI	Fan in
FO	Fan out
NOA	Number of all activities
NOSA	Number of structured activities
NOBA	Number of basic activities
ND	Nesting Depth
CW	Cognitive weight
WSDL	Web service description language
ANFIS	Adaptive neuro fuzzy technique
REST	Representational transfer
SOAP	Simple object access protocol
ESB	Enterprise service bus
SWWF	Stochastic well formed workflow
CFC	Control flow complexity
DOT	Depth of Tree
DIC	Data interaction complexity
XML	Extended markup language
WSC	Web service composition
QOS	Quality of web service

Chapter 1

Introduction

Service oriented architecture (SOA) is an architecture designed pattern to meet various business needs of the organization. It is a kind of software architecture, where design patterns consist of distinct pieces of logically encapsulated business functionalities called as web service. It provides interoperable, reusable and loosely coupled services to client. Web service consists of three main components, i.e, Service Provider, Service requester and Service Repository as shown in Figure 1.1.

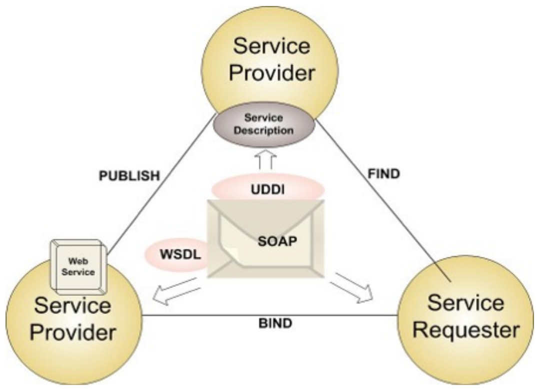


Figure 1.1: Service Oriented Architecture

Service Provider acts as server which provides the services to the client. It publishes the services to a registry and makes it available on the internet for the requests of the consumers. The service provider gets the request information from service requester and sends the response information to the service requster as shown in Figure. 1.2.

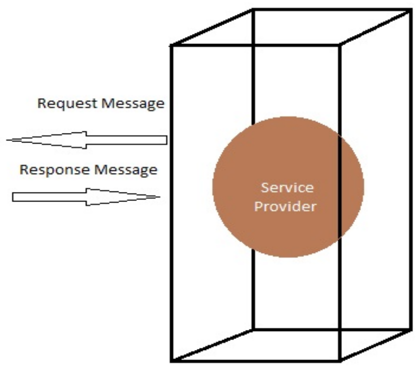


Figure 1.2: Service Provider

Service requester performs the service discovery operations and communicates with the service provider in order to exchange their messages using the standard protocol called as simple object access protocol (SOAP). Block diagram of the roles of service reqester is shown in Figure .1.3.

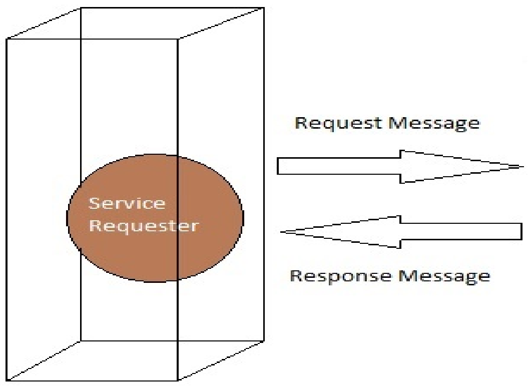


Figure 1.3: Service requester

Service Registry is the repository of web service. It is like a central storage device where all services are stored using the UDDI system. It can be get access by web service interface called as web service description language.

1.1 Web service

Web service is the basic build block of SOA. The important principle of web services are loose coupled, autonomous, and self-described. It is a modular application which can be published, located, and invoked across the web. Service interface is the one of the important key features of web service. It provides the machine to machine interoperability over the network. WSDL is the web service interface. It is an XML based language consist of message, port type, binding, operation and service. Web service has ability to integrate over the different organisational functionality. ESB act as the middleware through which different services are integrated.

1.1.1 Web service Composition

Service composition means composing two or more services as a single service. It mainly consits of two or more services as shown in Figure.1.4.

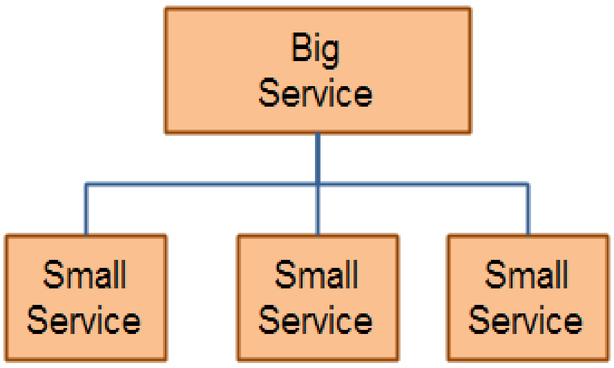


Figure 1.4: Web service composition

Orchestration of these web services is done through different compositional languages, i.e., BPEL (Business process execution language), BPML (Business process modeling language), and WSCL (Web service choreography language) etc. [1] and [2].

1.1.2 Business Process Execution Language

In this study, web services are composed using the BPEL compositional languages. The Bpel has been prevailed as the de-facto standard for executable processes. BPEL process is designed to interact with the external entities using web service description language [1].

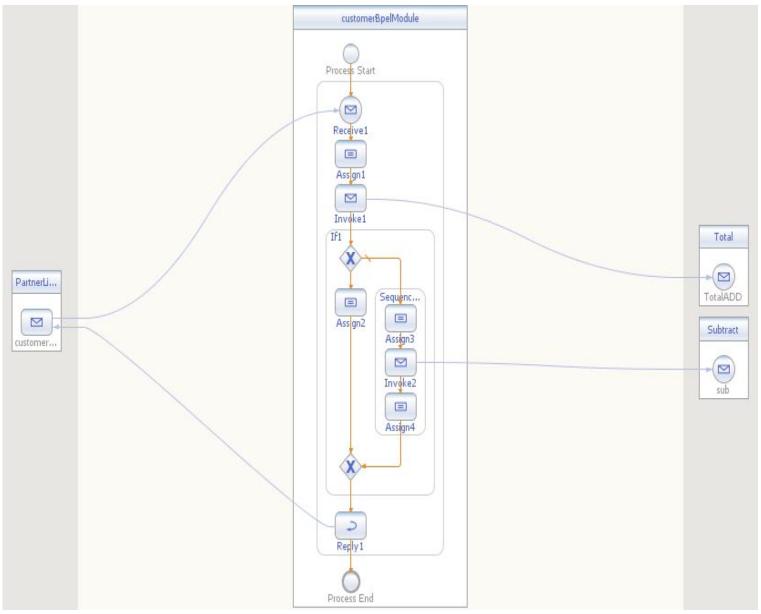


Figure 1.5: BPEL Diagram

BPEL is an xml based language described the service invocation, process invocation, and control structure of the business logic. The BPEL process consists of external-partners capabilities, variables, various handlers and activities. BPEL provides a mechanism for hierarchical and graph like structure.

BPEL activities are divided into two categories such as basic activities and structured activities.

a. Basic Activities

Basic activities are used for common tasks. These are responsible for calling and receiving messages, controlling process, and manipulating data. List of basic activities are given in Table1.1.

Table 1.1: Different Basic Activities of BPEL Process

Basic Activities	**Functionalities**
$< invoke >$	invocation of services
$< receive >$	input variable received from client
$< replay >$	output send to the process
$< assign >$	changing the data variables
$< faulthandelers >$	exceptions handler
$< wait >$	process wait for while
$< terminate >$	quit the entire process

b. Structured Activities

Execution constraint of different business process is defined by structured activities. These activities are providing the way of control flow and path in which data are executed. It is more complex and describes the flow of process by structuring basic activities. List of structured activities are given in Table 1.2.

In this work Open ESB v2 tool is used for designing the BPEL process. The design of BPEL is carried out using a graph like structure which can represent different web service invocation and condition in a graphical manner. Design part of BPEL can automatically generate the BPEL source code in an open ESB tool. The generated source code, is used to compute the different complexity values of web service composition.

Table 1.2: Different Structured Activities of BPEL Process

Structured Activities	**Functionalities**
$< sequence >$	sequential execution of activities
$< switch >$	specify conditional behavior
$< while >$	working iterative way.
$< pick >$	waits till one event in a set of events occur
$< flow >$	Used for parallel execution

1.2 Motivation

In traditional software like object oriented programming, there are various metrics are available for analysis of complexity of programs and quality assessment. Software industries often considers the use of different metrics for analysis of effort as well as quality of software. But in SOA, there are no specific metrics, which can help to conclude that web process is more or less complex. Hence it is necessary to compute different web process complexity metrics.

The need for multiple numbers of web service invocations indirectly increases the amount of complexity of web service composition. The complex web process is difficult to maintain, less reliable and of high risk [9]. Web process becomes more complex, if there is no limitation or restriction for a number of web service compositions. When a large number of web services are composed, then it increases the invocation, data complexity and flow complexity etc.. So it is necessary to analyze the different complexity parameters of web process. Effective complexity analysis of web process makes it easy to understand and flexible to design.

1.3 Literature Survey

In the year 1988, Weyuker proposed the method for analysis of software complexity. Author can evaluated the CFC in terms of Weyuker's properties [10]. Author can used the traditional software for calculating the control flow complexity. It can further improved by Cardoso, control flow complexity is calculated in terms of web process [4] and [7].

Gruhn and Laue provide different business process model (BPM) complexity metrics to measure complexity as shown in Table 1.3 [8]. These metrics are commonly used in terms of traditional software.

Table 1.3: Overview of process complexity metrics

Software Complexity Metric	**Corresponding BPEL complexity Metric**
Lines of Code	Number of basic activities (NOA) Number of all activities (NOAC)
Cyclometic Number	Control flow complexity (CFC)
Nesting Depth	Nesting Depth (ND)
Cognitive weight	Cognitive weight tailored for BPEL
Fan-in	Number of process invocations
Fan-out	Number of service invocations

Petri net based web process was proposed by R. Hamadi and B. Benatallah [3]. This approach has also been used for calculating the control flow complexity of web service composition utilising the model of BPEL [6]. But disadvantage in this approach was that it does not consider the structure of the source code.

E. Rolon et al. have proposed several metrics for business process in web service composition. Their metrics are an extension for modeling and evaluation of the software process. Reijers and Vanderfeesten have also proposed different metrics for computing the process cohesion and process coupling metric [2].

Misra and Misra (2004) have proposed an cognitive complexity. Cognitive complexity measures in terms of Weyukers property. According to this aaproach cognitive weight is the important parameter for software complexity measures and established the cognitive complexity as a well structured one [1].

Sha Jing and Du Yu-yue proposed an approach to give quality of service. The service composition developed based on a decomposing algorism and the numerical analysis. Author can use the stochastic well-formed workflow (SWWF) models of web service composition. This approach never considered any metrics for performance analysis [12].

Operational research technique for the performance analysis is provided by the book "Capacity Planning for Web Services: metrics, models, and methods" [11]. This book provide the performance and reliability measurement. It discussed in-depth about the workload and performance modeling.

Song Juan and Wang Hao proposed an formal model for quality anlysis of BPEL process. Queueing Petri Nets [1-2] formal modeling technique is used. Simulation tool is used for simulation of service composition, finally analyze the different performance indicators [13].

QoS(Quality of Service) of web services in terms of performance, reliability, and availability becomes the key issue when web services model complex. **Heejung Chang, and Hyungki Song** proposed a simulation-based framework. The web service composed based on their quality of web service. Author mainly concentrate on the quality as an important factor for web service composition as it is changes the humen perspective [14].

Zhangxi Tan, and Chuang Lin discussed five basic structures of web service flow: sequential, parallel, conditional, loop and mutex. Author can analyze the performance of a web service flow management system using this five flow control [15].

1.4 Objective of Research

The main objectives find from the motivation to work in web service composition are :

- Complexity metrics: Different complexity metrics are modeled using various business processes.
- Quality Model: Quality model using business process complexity metrics, has to be designed.
- In order to design the quality model, fuzzy logic is to be applied.
- Performance analysis of design models using different performance evaluation parameters.

1.5 Organization of the Thesis

The rest of the thesis is organized as follows:

1. Chapter 1: In this chapter basic concept of SOA, web service, web service composition are discussed. Motivation of the research work is described here. Different literature provides the different concept about the complexity and quality of web service is described here.
2. Chapter 2: Complexity metrics are modeled using the business processes.
3. Chapter 3: Quality model is proposed using the different complexity metrics.
4. Chapter 4: Fuzzy logic is considered to develop the quality model.
5. Chapter 5: Performance of quality model is evaluated by different performance evaluation parameter.
6. Chapter 6: This chapter covers the conclusion of this study.

Chapter 2

Business Process Complexity metrics

Meteric is the unit of measurement. Metrics are widely used in software industry for different quality analysis of software. Different metrics are used to analyze the software performnace, reliability and other quality attributes. Complexity metrics have been widely used predicting the error rate, estimating maintenance costs and mainly used in re-engineering of software. In traditional software like object oriented program large number of metrics is already defined. CK metrics, McCabe's Cyclomatic complexity etc. are widely used in different purpose like effort cost estimation, quality analysis, fault prediction etc.. But in the service oriented software very few metrics are defined. In this chapter different proposed metrics are defined and compare to the existing model.

2.1 Business process

Business process are based on work-flow, consisting of set of input, output, control structure, external invocation etc.. In this work seven different complexity metrics are defined. The different complexity metrics are explained below:

2.1.1 Control Flow Complexity [CFC] Metric

In traditional software, CFC is an important metric for measurement of complexity of software. BPEL model is depends upon the splits, joins, loops [5].

The working of algorithm for computing the CFC depends upon the number of independent paths in a structured tree, which is a type of control flow graph of BPEL process, as indicated in Algorithm 1.

Algorithm 1 BPEL Process CFC's algorithm

```
1: procedure Complexity($S - tree$)
2:     while $Independentpath \neq 0$ do
3:         $a \leftarrow COMPUTEPATH(Independentpath)$
4:         $b \leftarrow b + a$
5:         $path \leftarrow path + 1$
6:     end while
7:     $CFC \leftarrow b/path$
8:     return $CFC$
9: end procedure
10: procedure Computepath($Independentpath$)
11:     for $i \leftarrow 0, Pathlen$ do
12:         $s \leftarrow s + p[i] * weight$
13:     end for
14:     return $s$
15: end procedure
```

This algorithm mainly depends upon the structure tree graph also called as control flow graph. The structured and basic activities are represented by rectangle and condition is represented by diamond. Each activity and condition are assigned with some weightage are shown in Table.2.1 [4].

Control flow complexity (CFC) provides the independent path of execution. If cfc value is high then it indicate that the program response time is high and less

Table 2.1: COMPLEXITY WEIGHTS OF BASIC LOGIC STRUCTURES

Type Name	**Basic Structure**	**Weight**
Branch	if-else	2
	switch	3
	pick	3
	merge node	2
Iteration	while	3
	repeatUntil	3
	forEach	3
Concurrency	flow	4
	join node (flow)	4
Service Invocation	invoke	2
	reply node	2
Interrupt	exception handler	3
	event handler	3

reliable for web service. It affects the performance of web service composition.

2.1.2 Data Interaction Complexity [DIC]

In service oriented architecture, web services communicate with each other in Bpel process. The interaction of data between the web services is done through the protocol called simple object access protocol (SOAP). Structure of SOAP is given below in Figure 2.1.

A SOAP is a message oriented protocol containing the Envelope, Header, and Body. Envelope defines the XML documents as a SOAP message. The header contains information about the message and body contains call and response information. SOAP message element can be divided into two types such as basic primitive type element and complex type element. The basic type refers to the

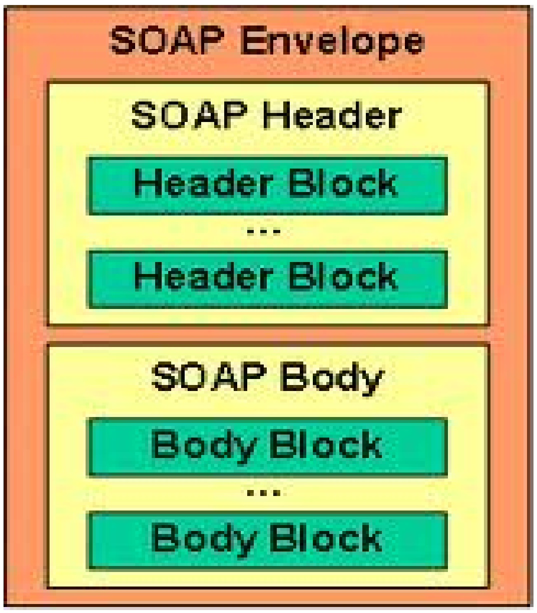
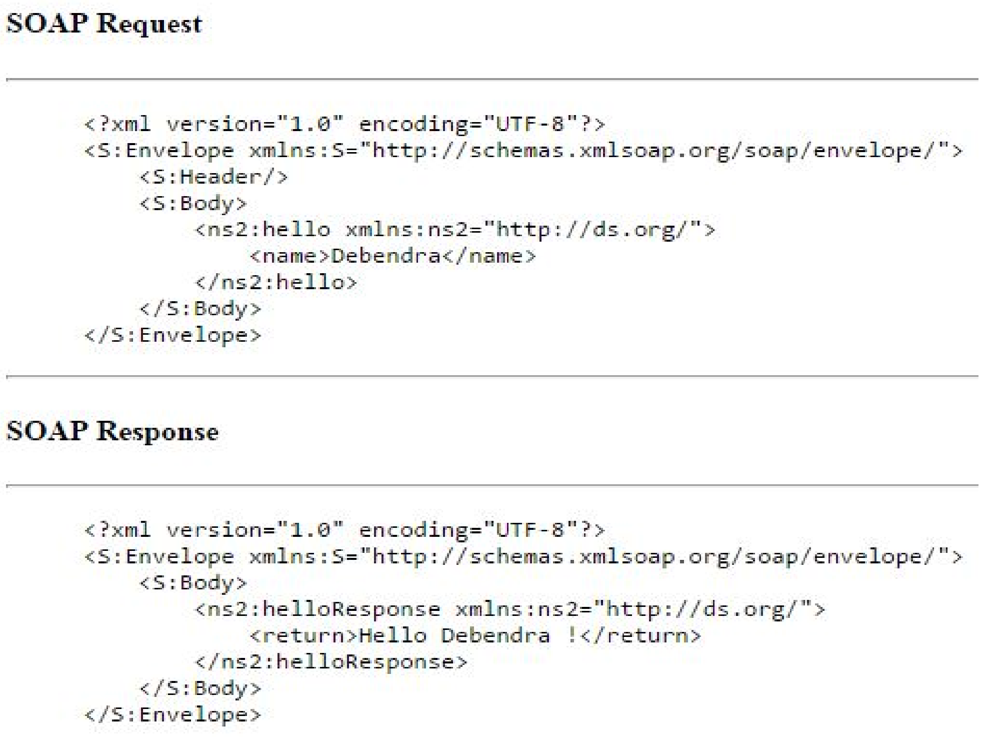


Figure 2.1: SOAP Structure



SOAP Request

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
    <S:Header/>
    <S:Body>
        <ns2:hello xmlns:ns2="http://ds.org/">
            <name>Debendra</name>
        </ns2:hello>
    </S:Body>
</S:Envelope>
```

SOAP Response

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
    <S:Body>
        <ns2:helloResponse xmlns:ns2="http://ds.org/">
            <return>Hello Debendra !</return>
        </ns2:helloResponse>
    </S:Body>
</S:Envelope>
```

Figure 2.2: Soap request and response message

primitive data types such as integer, float and string. The complex types refer to the composition of basic types.

Data interaction complexity can compute the complexity of data in web service composition. Bpel process contains different web services, which can interact with each other shown in source code as given in Figure. 2.2.

For calculating the data interaction, complexity value can be found from following equation:

$$DIC(p) = \sum DIC_{complex}(elem) + \sum DIC_{basic}(ele) \tag{2.1}$$

$$DIC_s(WSC) = \frac{2 * \sum_i DIC(inv_i)}{N_{ws} * (N_{ws} - 1)} \tag{2.2}$$

where $DIC(p)$ is Data Interaction complexity of BPEL process, $DIC(inv_i)$ Data inteaction complexity of ith service invocation.

$DIC_{basic}(elem) = 1$, if element is basic type data.

$DIC_{complex}(elem) = \sum DIC_{basic}(elem)$

2.1.3 Fan In [FI] and Fan Out [FO]

The number of ways the process is initiated is called Fan In (number of process invocations). Petrinet model is used to design the business process. In Figure 2.3, process1, process2 and process3 represent the process invocation and service1, service2, service3 represent the service invocation.

The number of ways the process is initiated is called Fan In (number of process invocations).

$$FI(P) = \sum_{a \epsilon P}^{n} FI(a) \tag{2.3}$$

where FI(P) is fan in of BPEL process and FI(a) is number of process invocations.

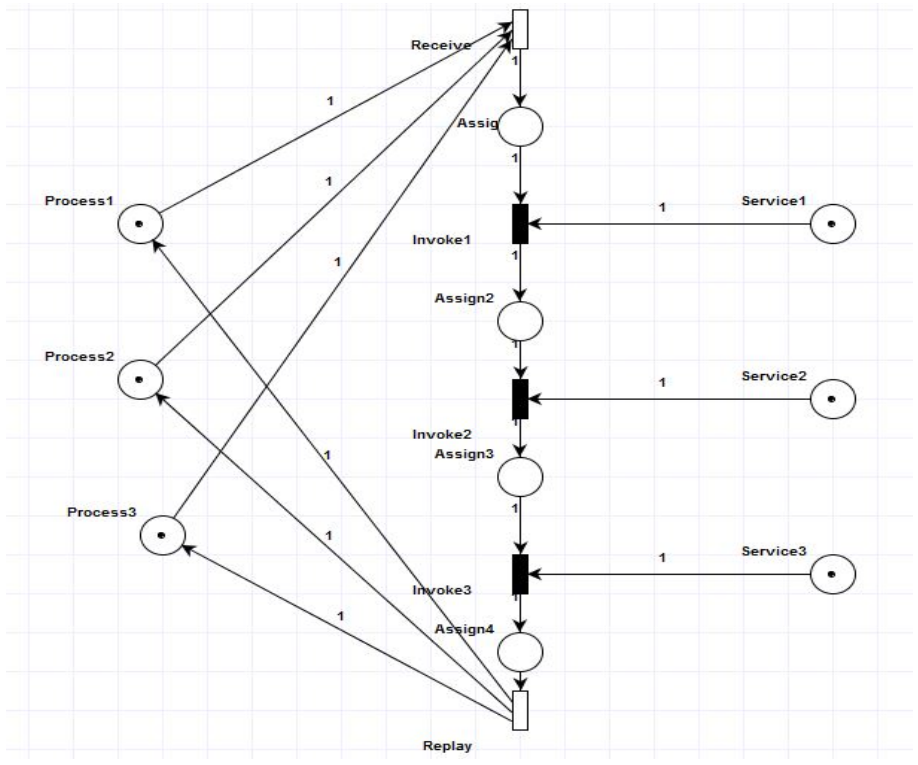


Figure 2.3: Petrinet model for process and service invocation

The number of ways service is invoked is called Fan Out (Number of service invocations).

$$FO(P) = \sum_{a \epsilon P}^{n} FO(p) \tag{2.4}$$

where FO(P) is fan out of BPEL process and FO(a) is number of service invocations.

2.1.4 Depth of Tree [DOT]

DOT basically depends upon the structure of BPEL process tree. BPEL tree is a kind of XML tree, generated by the use of BPEL2PNML jar file.

The nested condition in BPEL source code increases the basic activities and structured activities. The depth of the condition with respect to different activities is computed by DOT.

DOT is the one of the important complexity metrics of web service compositions. If DOT value is high, it means that the number of activities is more and it gets affected by other complexity values.

2.1.5 Nesting Depth [ND]

According to Cardoso, the nesting depth of an action is the number of decisions in the control flow that are necessary to perform the action [5].

ND (P) indicates as to how many structured activities are nested to the deepest activity in a process P, where P is a BPEL process.

2.2 Result and Analysis of Different Complexity Metrics

A case study on 'Library system' has been considered to compute the metrics and analyze as to how these metrics get affected for the non functional properties of software. The result of above metrics is given below:

2.2.1 CFC

The control flow graph is drawn from the BPEL model shown in Figure. 2.4

Different Path in CFG of Library BPEL process:

Path 1 $r1[1] \rightarrow a1[1] \rightarrow i1[2] \rightarrow c1[2] \rightarrow a3[1] \rightarrow i2[2] \rightarrow c2[2] \rightarrow a4[1] \rightarrow i3[2] \rightarrow a5[1] \rightarrow a6[1] \rightarrow i4[2] \rightarrow a7[1] \rightarrow c3[2] \rightarrow c4[2] \rightarrow r2[1]$

Path 2

$r1[1] \rightarrow a1[1] \rightarrow i1[2] \rightarrow c1[2] \rightarrow a3[1] \rightarrow i2[2] \rightarrow c2[2] \rightarrow a8[1] \rightarrow c3[2] \rightarrow c4[2] \rightarrow r2[1]$

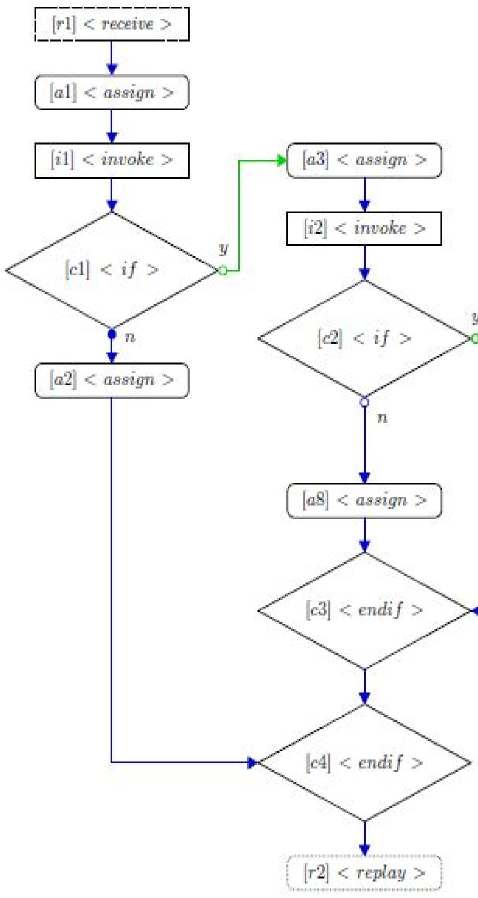


Figure 2.4: Control flow graph of Library BPEL process

<u>Path 3</u>

$r1[1] \rightarrow a1[1] \rightarrow i1[2] \rightarrow c1[2] \rightarrow a2[1] \rightarrow c4[2] \rightarrow r2[1]$

Hence complexity value can be computed as:

$$C(p) = \sum_{i=1}^{n} C(n_i) \tag{2.5}$$

where n_i is the node in path p. If $C(WSC)$ is the complexity value of BPEL process, then it can be computed as:

$$C(WSC) = \frac{\sum_{i=1}^{n} C(n_i)}{|PathSize|} \tag{2.6}$$

Its value for the Library system can be found as:

$C(WSC) = \frac{C(p1)+C(p2)+C(p3)}{|PS|} = \frac{25+17+10}{3} = 17.33$

2.2.2 Data Interaction Complexity

Data interaction complexity of Library system can be found out as:

$DIC(Librarysystem) = 3$, $DIC(Checkrollno) = 2$, $DIC(CheckISBN) = 2$, $DIC(UpdateIssue) = 2$, $DIC(UpdateBalance) = 2$

$DIC(WSC) = \sum_{i} DIC(inv_i)$

$DIC(WSC) = \frac{(2+2+2+2+3)*2}{5(5-1)}$=1.1

The value of $DIC(WSC)$ indicates less data interaction; so it can be interpreted as a reliable one.

2.2.3 Fan In [FI] and Fan Out [FO]

Fan in indicates the process invocation as the Figure 2.5. It shows that the number process invocation is one and number of service invocation is four.

$FI = process1 = 1$ and $FO = service1 + Service2 + Service3 + Service4 = 4$

FI and FO is an important metrics for predicating the performance of web service composition. FO increases when the number of service interaction is more; so it is also dependant upon the reusability properties.

2.2.4 Depth of Tree

Depth of the activities in a Library BPEL process is shown in Figure. 2.7. DOT is computed by this BPEL tree, i.e, Tree Height = 6, shown in Figure. 2.6.

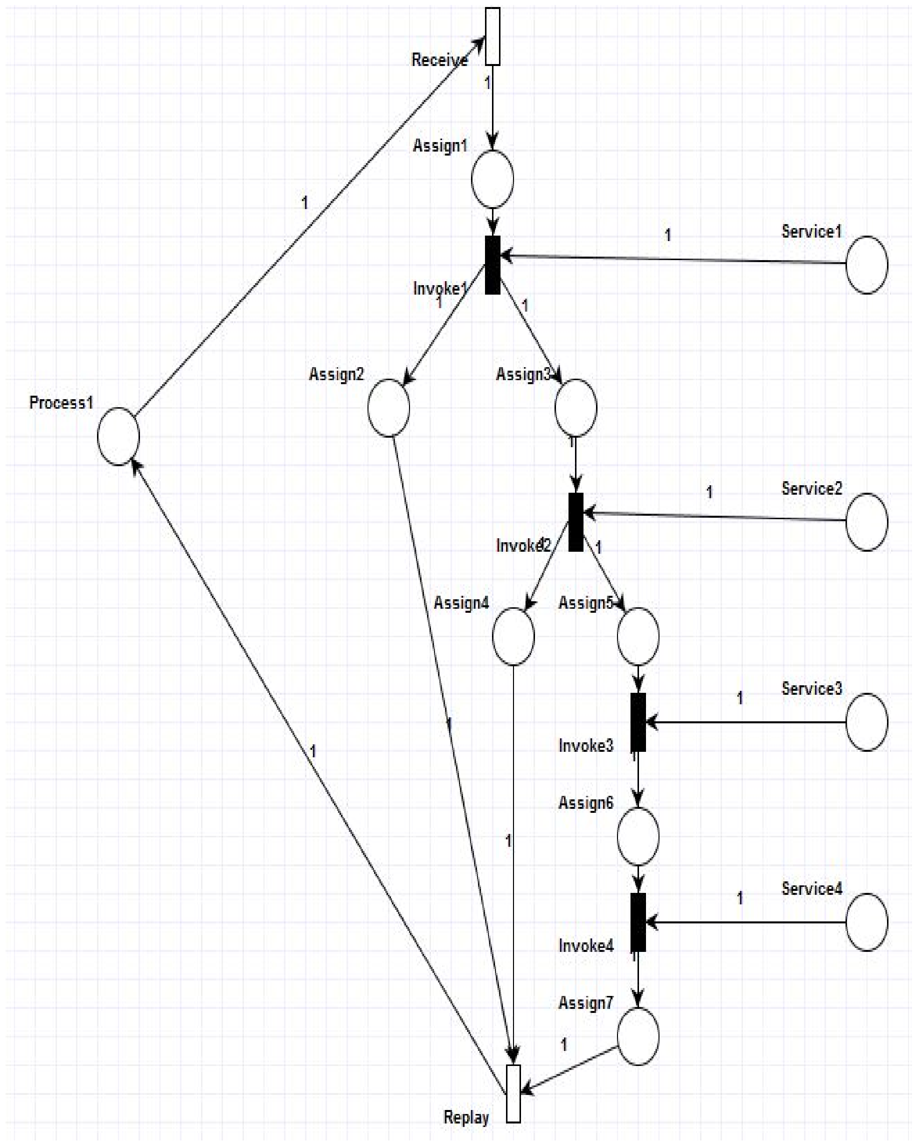
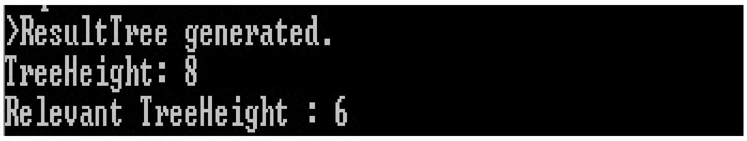
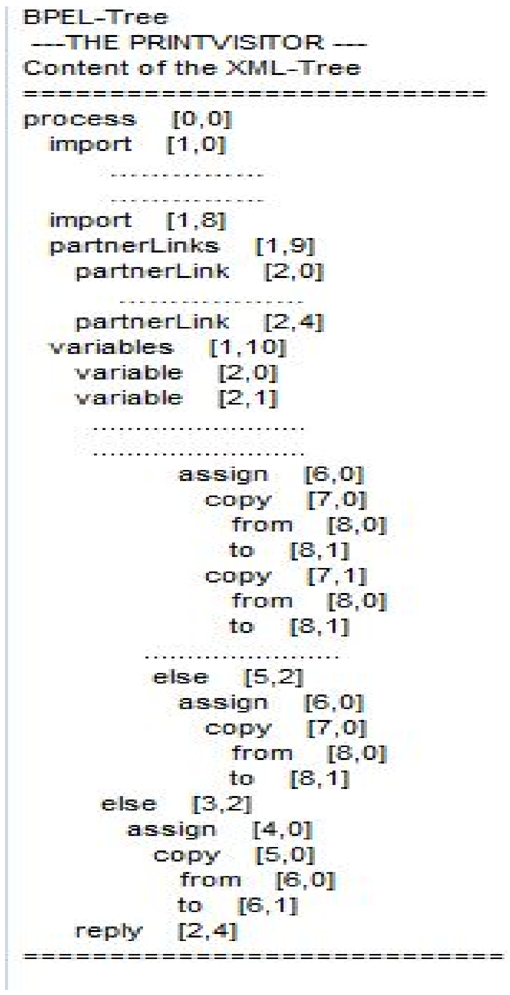


Figure 2.5: Petrinet model for process and service invocation of Library system



```
>ResultTree generated.
TreeHeight: 8
Relevant TreeHeight : 6
```

Figure 2.6: Result of DOT



```
BPEL-Tree
 ---THE PRINTVISITOR ---
Content of the XML-Tree
============================
process    [0,0]
  import    [1,0]
       ..............
       ..............
  import    [1,8]
  partnerLinks    [1,9]
    partnerLink    [2,0]
        ................
    partnerLink    [2,4]
  variables    [1,10]
    variable    [2,0]
    variable    [2,1]
      .........................
      .........................
             assign    [6,0]
               copy    [7,0]
                 from    [8,0]
                 to    [8,1]
               copy    [7,1]
                 from    [8,0]
                 to    [8,1]
          .......................
           else    [5,2]
             assign    [6,0]
               copy    [7,0]
                 from    [8,0]
                 to    [8,1]
      else    [3,2]
        assign    [4,0]
          copy    [5,0]
            from    [6,0]
            to    [6,1]
    reply    [2,4]
=============================
```

Figure 2.7: BPEL Tree

2.2.5 Nesting Depth

Nesting depth of library system is dependant upon the number of nested path through which data passes into the process output.

In other words, Service4 input value is dependent upon the Service1 output variable so its ND is 3.

2.3 Comparision with Related work

In this section proposed model is compared with the different existing model.

2.3.1 Control flow complexity

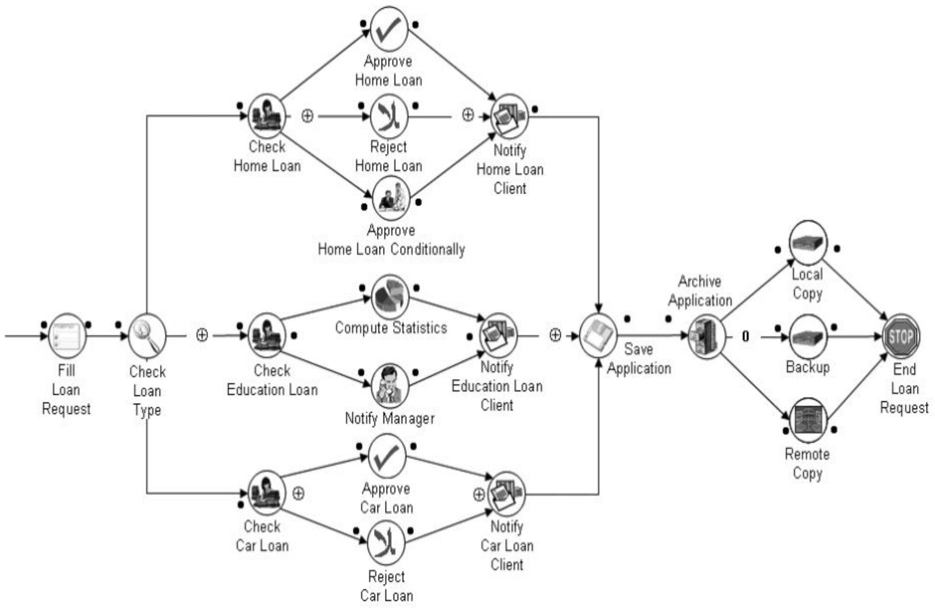


Figure 2.8: Loan Appliaction Process

Cardoso proposed a control flow complexity metric using the work flow diagram shown in Figure 2.8. The complexity of processes i.e. CFC is formulated by XOR, OR and AND-split constructs [10]. The formula used to calculate the control flow complexity is given below.

$CFC(P) = \sum_{i \in (XOR-splitsofP)} CFC_{XOR-split}$ (i) + $\sum_{i \in (OR-splitsofP)} CFC_{OR-split}$ (j)+ $\sum_{i \in (AND-splitsofP)} CFC_{AND-split}$ (k)

Chengying Mao proposed CFC based on petrinet model shown in Figure 2.9. The formula used to calculate the control flow complexity is given below.

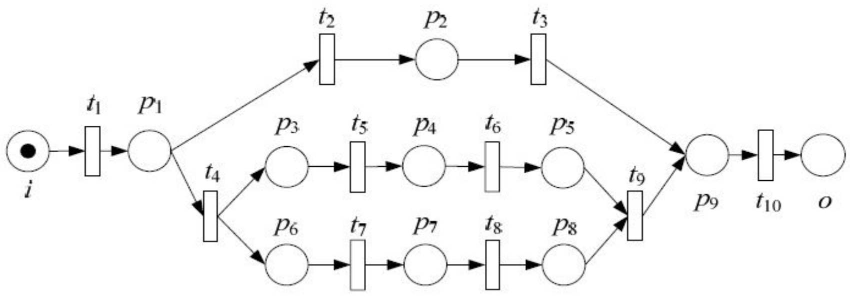


Figure 2.9: Petrinet model of Business Process

NP $=|P|$, where P is the place set in Petrinet, NT $= |T|$, where T is the transition set in Petri net and F is a set of directed arcs in Petri net.

$$CFC = |F| - |P| - |T| + 2 \tag{2.7}$$

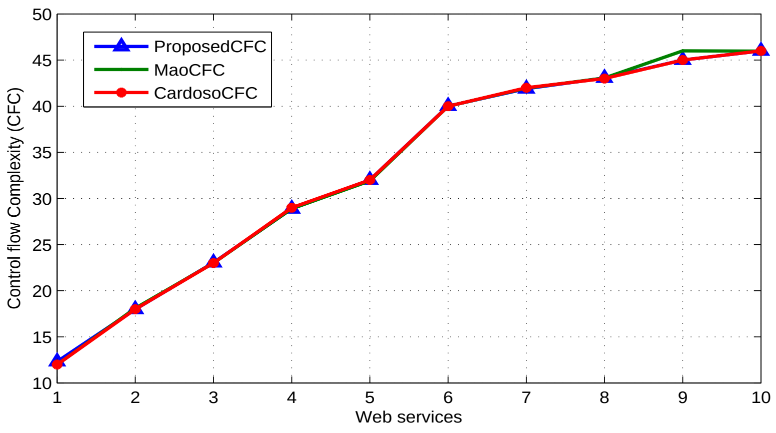


Figure 2.10: Comparision of proposed CFC with previous model

Above approach is used to compute the different business processes and finally compared with the proposed model shown in Figure 2.10. Proposed model is approxmately same value of CFC with the other model; but this model provides the efficient way of calculation of CFC as compared to other model as there is no need to design any model for calculation of CFC. It can directly be calculated by its source code of 'Bpel'.

2.3.2 Data interaction complexity

DIC metrics proposed by Cardoso, is compared with proposed DIC metrics shown in Figure 2.11.

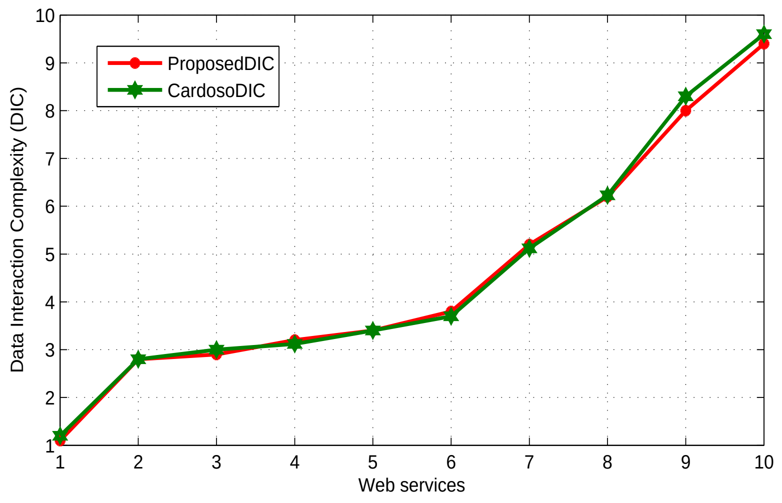


Figure 2.11: Comparision of proposed DIC with previous model

2.4 Summary

In this chapter, different complexity metrics are discussed. CFC is the standard metric for software performance analysis. A case study of library system is considered to compute the different complexity metrics and finally compare with the CFC and DIC value of with previous model. The proposed model is easily used to compute the complexity metrics. Depth of tree is used to compute the coupling of web service. These metrics can be used to calculate the quality of business process.

Chapter 3

Quality Model of Business Process

3.1 Introduction

There are various articles on quality assessments on traditional software but for SOA based software, there is less number of articles on quality assessment. SOA based software mainly consists of web services or composed web service, which affect the service quality. Quality of web service composition mainly depends upon different complexity metrics, i.e., activity complexity, structural complexity, control flow complexity etc..

Web Services (WS) are the technology to realize the services of SOA. They are used to implement the functional aspects of business processes, which in brief define the input/output behaviour of a component. In addition, in many business domains it is crucial to fulfil non-functional requirements. A non-functional requirement or Quality of Service (QoS) attributes help to specify as to how a component is supposed to behave in a complex environment. QoS attributes are mostly robustness, security, performance, scalability etc.. The different metrics of web service are closely associated with these high level QOS attributes. In this study various models have been proposed for quality assessment using different complexity metrics.

3.2 Model Development

The methodology used for development of hierarchical Quality model consists of three steps, as discussed below.

3.2.1 Identification of quality attributes associated during design

Software quality is multifaceted concept. In order to achieve the right quality, few models are available in literature based on a set of attributes, characters and metrics.

One of the earliest software quality models follows ISO/IES 9126 attributes. This quality aspect was fixed as per international standard. It has been replaced by ISO/IEC 25010:2011. A quality model is being composed of five characters (some of which are further subdivided into sub characters), that relate to the outcome of interaction when a product is used in a particular context of use.

In this paper five high level quality attributes are used for quality analysis of web service composition. This type of quality models are used to relate to various business complexity metrics in order to achieve the quality of web service composition.

3.2.2 Relationship between the Quality Attributes and Business complexity metrics

Quality of web service composition is computed using different complexity metrics. Five quality attributes are interrelated with the business complexity metrics.

Reusability

The main design principle of SOA is the reusability of business process in different domain. Business process reusability is defined as the extent to which a Business process (i.e., composite service) can be reused in other contexts, organizations, or SOA solutions with minimal effort and change. This means that an architect should analyze whether the given composite service can be reused in other business processes or domains or not. By reusability it is understood that every identified, specified, realized, and implemented BPEL process should be reused in different possible service oriented solutions.

Business process consists of a list of web services. If web services are composed with less interdependency then it is easy to re-use, i.e, reusability depends upon the control structure of Business process. Data interaction is the one of the important properties of reusability. Business process are defined in terms of basic activities.

Usability

Usability is an important factor during the software development life cycle. It provides how easily software can use. It can calculate the time to accomplish a particular task [19].

Business process is a kind of flow structure which consists of different activities and invocation of different web services. Usability of business process means consistency in the flow structure, i.e, fewer Consistencies means easy to use or easy to understand the structure. It mostly depends upon the control structure of the business process, i.e, CFC, ND, NOAC [18].

Reliability

Reliability is one of the import properties that are used to measure the quality of software. Reliability means mean time to failure. Reliability is widely used in the software industries as an important parameter for effective analysis of software fault analysis. Web service composition is consisting of split and join that can create process in deadlock and inconsistence state. This type of condition is handling by reliability analysis of web service. It is a user-oriented quality factor that relates to the system operation. A system without faults is considered to be highly reliable [22]. Reliability of web service composition depends upon the following metrics of business process such as CFC, ND, NOSA, FO [20].

Performability

Performability is the one of the important quality attributes of web service composition. Performability of webservices refer to means speed, efficiency, throughput etc.. Performance mainly depends upon the response time, i.e, low response time means high performance and vice versa.

Performance of web service mainly depends upon four metrics, i.e, NOA (Number of activity), Cognitive weight(CW), FO (Fan out), Number of strucutred activity (NOSA).

Maintainability

Maintenance refer to ability of a business process to be retained in its original form, and restored to that form in case of a failure. Maintenance depends upon the coupling and cohesion factor of web service composition [21].

Cohesion refers to the degree to which the elements of a module belong to other modules. Thus, it is a measure of how strongly they are related with each piece of functionality expressed by the business process. Coupling is the degree of interdependence between two modules. High cohesion with low coupling is preferred for Business process maintainability. Maintainability is calculated by the metrics such as CFC, FO, and NOSA; as they depend upon the aspects of cohesion and coupling [21].

3.2.3 Design Equation using Dependent and Independent Variables

The goal of this study is to explore the relationship between business process complexity metrics and different quality attributes. Table 3.1 interrelates the quality attributes with complexity metrics. Reusability is a function of FI, FO, CFC, and

NOSA can represented as shown in the following equation:

$$Reusability = f(FI, FO, CFC, NOSA) \tag{3.1}$$

Like wise other attributes interrelate to the complexity metrics as shown in Table 3.1.

Table 3.1: Relation of Complexity Metrics to the Quality Attributes

Quality Attributes	NOA	NOAC	ND	CW	FI	FO	CFC	NOSA
Reusability	×	×	×	×	✓	✓	✓	✓
Usability	×	✓	✓	×	×	×	✓	×
Reliability	×	×	✓	×	×	✓	✓	✓
Performability	✓	×	×	✓	×	✓	×	✓
Maintainability	×	×	×	✓	✓	✓	✓	✓

3.3 Data Gathering And Analysis Techniques

The following section describes the dataset being used for design of the quality model. Data are normalized to obtain the value of accuracy, using different dependent variable and independent variable .

3.3.1 Empirical Data Collection

A case study from literature has been considered where 1,145 BPEL processes complexity metrics values are used [16] [17]. It includes ActiveVOS reference applications, IBM Industry Content Packs, Oracle Fusion Applications, and Oracle Application Integration Architecture. The 1,145 collected BPEL processes are from various application domains, mainly from Customer relationship management (CRM), service management and operations (SMO), human resource management (HRM), resource management and operations (RMO).

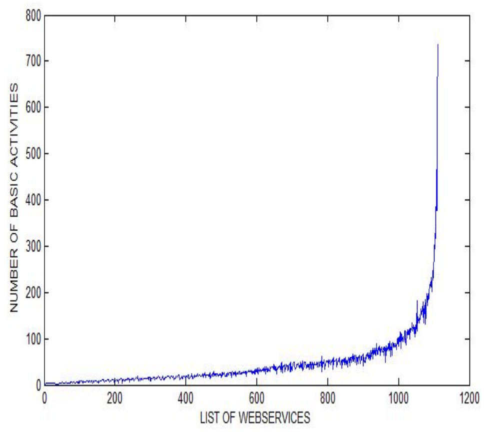


Figure 3.1: Number of Basic activities

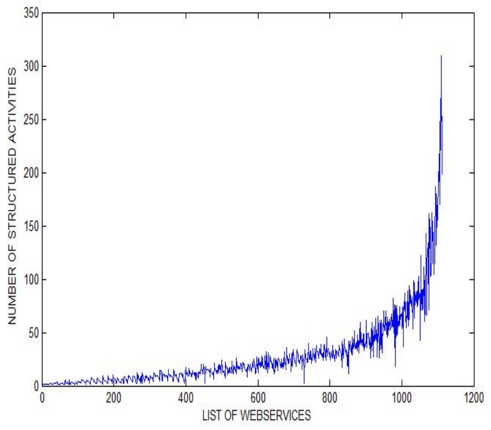


Figure 3.2: Number of structured activity

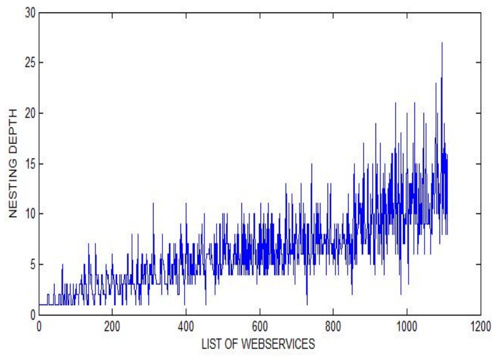


Figure 3.3: Nesting Depth

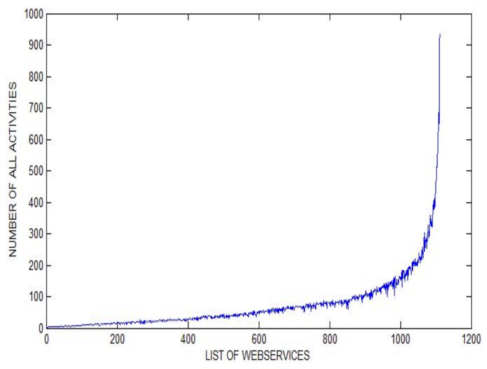


Figure 3.4: Number of all activity

3.3.2 Data Analysis

Flow design increases with the increase of a number of basic and structured activities. Moreover, processes with high NOAC increases the degree of process robustness. There is a strong correlation between NOA and NOAC. It is observe that both metrics have a strong correlation to CW and FO while their correlation with CFC is weak. The value of CW metrics is mainly depends upon the basic and structured activities. For this reason the CW has a strong correlation with NOA and even stronger with NOAC. CW also gives relatively high values for activities measured with FO. Therefore, a strong correlation between CW and FO is identified. Activities

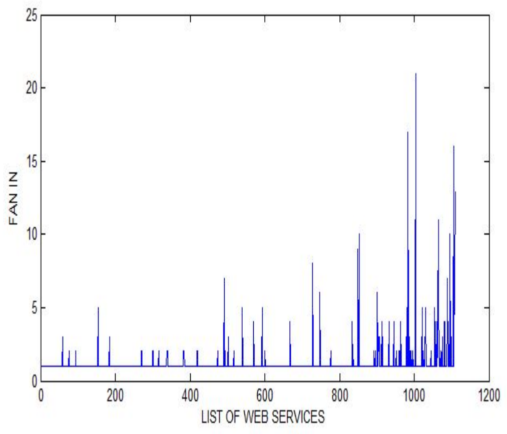


Figure 3.5: Fan in

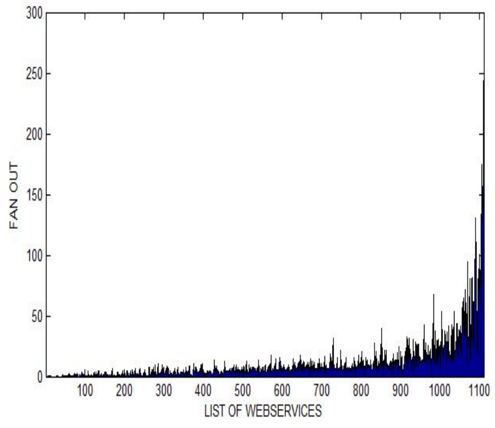


Figure 3.6: Fan out

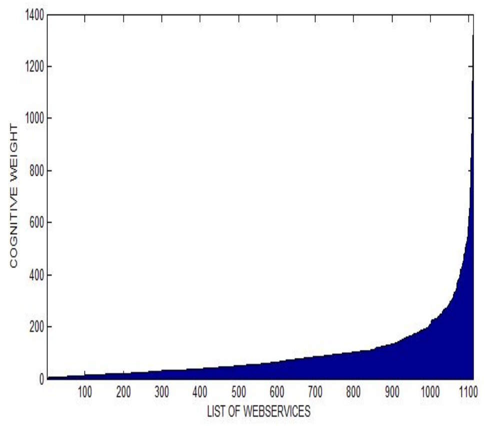


Figure 3.7: Cognitive weight

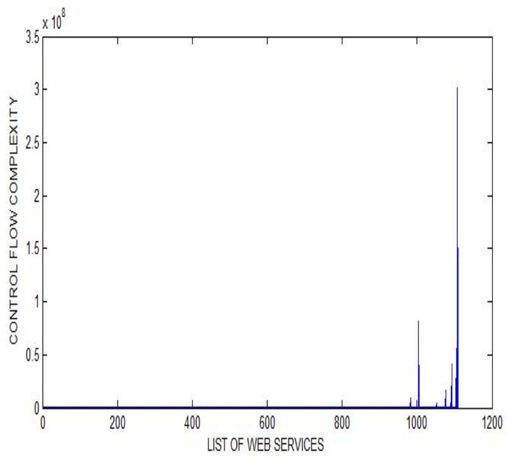


Figure 3.8: Control flow Complexity

with the higher process has, the greater chance of invoking (activity measured by FO). Therefore, FO and NOA correlation is strong. Similarly, the correlation between NOAC and FO is also strong. CFC is high in processes with complex pick and if activities.

Process nesting depth directly depends on the number of structured activities; this is confirmed by the correlation of ND and NOAC. The calculated process complexity values indicate that some metrics (e.g., NOA and FO or CW and NOA) are highly correlated and that they measure similar relative values. On the other

Table 3.2: Correlations of Complexity Metrics

	NOA	**NOAC**	**ND**	**CW**	**FI**	**FO**	**CFC**	**NOSA**
NOA	1	0.989	0.590	0.990	0.426	0.919	0.271	0.929
NOAC	0.989	1	0.653	0.996	0.378	0.885	0.250	0.972
ND	0.590	0.653	1	0.630	0.038	0.409	0.039	0.727
CW	0.990	0.996	0.630	1	0.419	0.912	0.259	0.962
FI	0.426	0.378	0.038	0.419	1	0.553	0.327	0.284
FO	0.919	0.885	0.409	0.912	0.553	1	0.269	0.792
CFC	0.271	0.250	0.039	0.259	0.327	0.269	1	0.204
NOSA	0.929	0.972	0.727	0.962	0.284	0.792	0.204	1

hand, CFC and FI are fairly independent of all other metrics, indicating that they measure unique aspects of the process complexity. Based on calculated Pearsons correlation coefficients for all eight metrics is given in Table 3.2.

3.4 Summary

In this chapter, the high level quality attributes are observed to have interrelated with the design complexity metrics. The complexity metrics are used to develop the empirical function. Empirical data are collected from the different literature and analysed. From the dataset it concludes that all complexity metrics are interrelates with each other. This data set is used to verify the model. Different soft computing technique like fuzzy logic and neuro fuzzy technique are used to develop the model.

Chapter 4

Quality assessment of web service composition using Fuzzy logic

4.1 Fuzzy Rule Based Approach

Fuzzy logic is a method for computing the uncertainties of any problem arising due to vagueness [24]. It consists of three different phases. The flow diagram of Fuzzy logic is given in Figure.4.1.

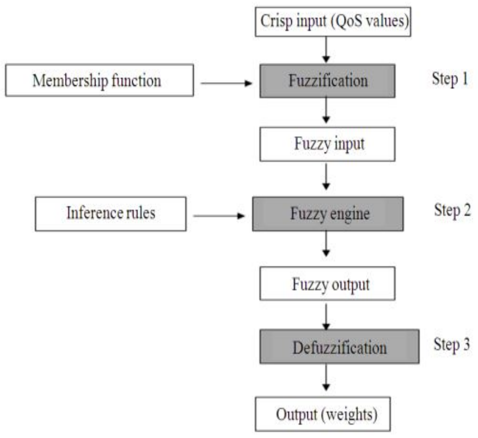


Figure 4.1: Fuzzy logic flow diagram

Fuzzification is the process of transforming the crisp values into linguistic terms of fuzzy set. A particular membership function is developed to associate a grade to each linguistic term. The crisp inputs are identified and the degree to which these inputs belong to each appropriate fuzzy sets using membership function, are determined. Triangular function is being considered for calculating degree of membership function for input as shown in following equation.

$$\mu_A(z) = \begin{cases} 0 & z \leq k \\ \frac{z-k}{s-k} & k \ < z \leq s \\ \frac{n-z}{n-s} & s \ < x \ < n \\ 0 & n \geq z \end{cases} \tag{4.1}$$

where $\mu_A(z)$ is called the membership degree of z in the fuzzy set $\tilde{A}$.

The triangular function is defined by a lower limit 'k', an upper limit 'n', and a value 'm',where k < s < n.

The membership function of this output variable uses the Gaussian membership function with the central value as 'm' and standard deviation as k>0.

$$\mu_A(z) = e^{\frac{-(z-s)^2}{2k^2}} \tag{4.2}$$

The result of membership function is fed as input to fuzzyengine and further applied on the fuzzy rule from the expert domain.

The last phase of fuzzy-inference system is "defuzzification', which uses the center of gravity technique. It is expressed as:

$$z = \frac{\sum_{i=1}^{n} m_i w_i}{\sum_{i=1}^{n} m_i} \tag{4.3}$$

where: z = The defuzzified output

m_i = the membership of the output of each rule

w_i = the weight associated with each rule

4.1.1 Model design using Fuzzy logic

Quality model is designed using the fuzzy logic, based on certain input and output paramters. Reusability is observed to be dependant upon the NOSA, FI, FO, CFC business complexity metrics. These metrics are used to design the model. It consists of four input and one output parameters as shown in Figure. 4.2, Figure.4.3 and Figure.4.4.

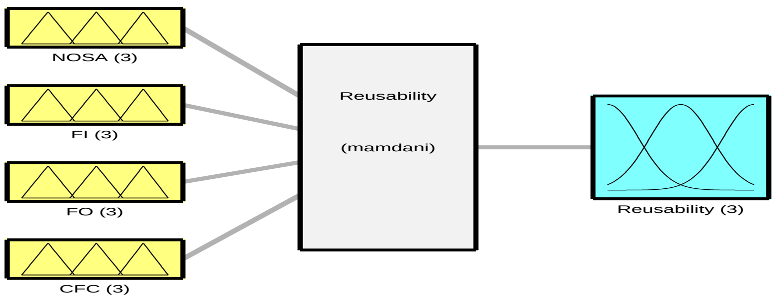


Figure 4.2: Inference Engine of Reusability

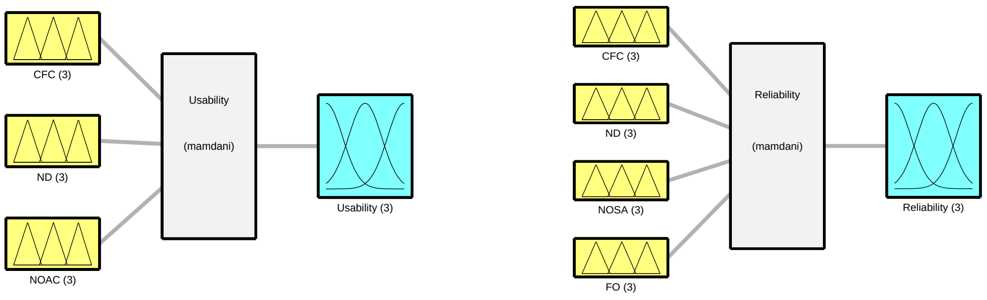


Figure 4.3: Inference Engine of Usability and Reliability

In the first phase, the crisp values are transformed into linguistic terms. Quality attributes use the three linguistic terms, i.e, Low, Medium, High. A membership function is developed to associate a grade each linguistic term as shown in Fig.4.5. The triangular membership function is used to design the mebership function for

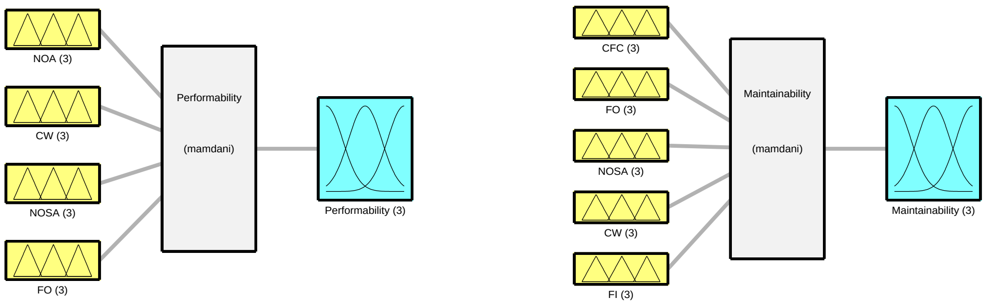


Figure 4.4: Inference Engine of Performability and Maintainability

input variable with different ranges as given in Table. 4.1. The crisp inputs are identified and the degree to which these inputs belong to each appropriate fuzzy sets using membership function, are determined.

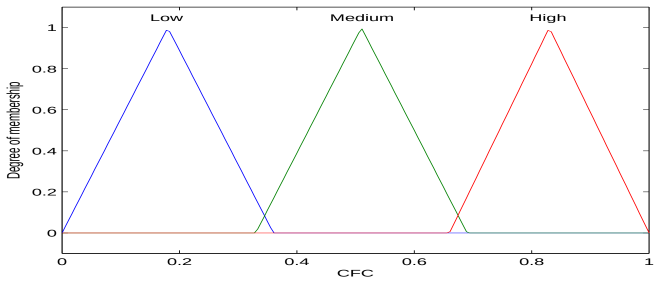


Figure 4.5: Membership Function of Control flow Complexity

Table 4.1: Membership functions and their ranges for input parameters

Membership function	**Range**
Low	0 - 0.36
Medium	0.33 - 0.69
High	0.66 - 1

The output variables are assigned to three linguistic phrases such as Low, Medium, High. The membership function of this output variable uses the Gaussian membership function. The membership function of output variable is shown in Figure. 4.6.

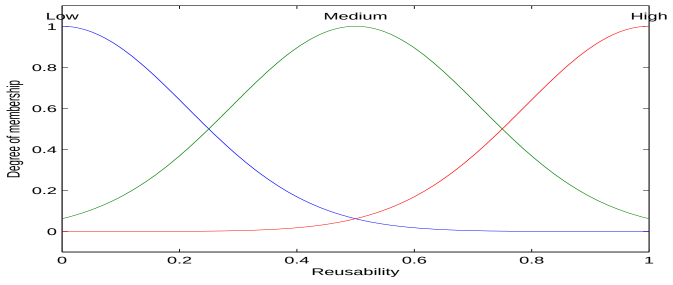


Figure 4.6: Reusability Membership function

The result of membership function is fed as input to fuzzyengine shown in Fig. 4.1 and further applied on the fuzzy rule. The rule is specified by analyis of dataset and from the expert domain shown in Table 4.2.

Table 4.2: Fuzzy rules specified in Risk probability fuzzy inference system

Rule No.	NOSA	FO	FI	CFC	Reusability
1	L	H	L	L	High
2	M	L	L	L	High
3	M	H	H	M	Low
4	M	M	M	H	Medium
5	M	H	H	M	Medium
6	H	H	M	H	Low
-	-	-	-	-	-
-	-	-	-	-	-
29	M	L	L	M	Medium
30	H	H	H	M	Low

An example of rule may be given as follows: **if** (NOSA is Low) **and** (FO is High) **and** (FI is Low) **and** (CFC is High) **then** (Resability is Medium).

Other quality attributes as shown in Figure. 4.2, 4.3, 4.4 are developed using the same approach and define the rules by analyzing the dataset and the output is finally computed by defuzzification mechanism using MatLab Tool. It can generate the output of the above system in Rule Viewer. The last phase of fuzzy-inference system is 'defuzzification', which uses the center of gravity technique.

4.2 Summary

In this chapter, fuzzy logic is used to design the quality model. All quality attributes are used as output variable in inference engine and all its complexity metrics are used as input variable. Fuzzy logic is the efficient technique for analyzing the vagueness problem having not an efficient algorithm for analysis. Here, rules are assigned to each model to compute the quality of web service composition.

Chapter 5

Performance Analysis of Different model

5.1 Performance Evaluation Parameters

Four performance evaluation parameters such as Mean absolute error (MAE), Mean absolute relative error (MARE), Root mean square error (RMSE) and Standard error of the mean (SEM) are very often considered for calculation of accuracy while using softcomputing techniques[23].

1. **Mean Absolute Error (MAE)** Mean Absolute Error is used to measure how close predictions to the eventual outcomes.

$$MSE = \frac{1}{n}\sum_{i=1}^{n} |y_i - y_i'| \tag{5.1}$$

2. **Mean Absolute Relative Error (MARE)**

   MARE is a used to measure how close forecasts or predictions are to the eventual outcomes.

$$MARE = \frac{1}{n}\sum_{i=1}^{n} \frac{|y_i - y_i'|}{y_i + 0.05} \tag{5.2}$$

3. **Root Mean Square Error (RMSE)**

   RMSE is a measure of the differences between values predicted by a model or an estimator and observed value.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - y_i')^2} \tag{5.3}$$

4. **Standard Error of mean (SEM)**

   The standard error of the mean is designated as σ_M. It is the standard deviation of the sampling distribution of the mean. The formula for the standard error of the mean is:

$$\sigma_M = \frac{\sigma}{\sqrt{N}} \tag{5.4}$$

   where σ is the standard deviation of the original distribution and N is the sample size.

5.1.1 Performance of Quality Model

The output of both the techniques are compared with the actual output using performance evaluation parameters. Comparision of both the design model is shown in Table. 5.1.

Table 5.1: Performance Evaluation of Different Quality Model

Quality Attributes	MAE	MMRE	RMSE	Std. Error
Reusability	0.0216	0.0255	0.1530	0.0265
Usability	0.0720	0.0390	0.3771	0.0395
Reliability	0.0306	0.0154	0.1749	0.0317
Performability	0.0729	0.0366	0.2733	0.0337
Maintainability	0.0345	0.0125	0.0325	0.0205

Chapter 6

Conclusion

This study intends to focus on different complexity metrics of Bpel process. When the result obtained for complexity metrics of Bpel process are compared with other approaches, it is observed that the result from the proposed approach provides the better result while computing the complexity value. Bpel Tree is the new approach for computing the Depth of the tree. DOT provides the depth of the process execution. Here Library system Bpel process is used to compute all the metrics and it can easily be compared with another approach.

Activity complexity and structural complexity are found as metric to measure the effort required for comprehending a piece of software. Limitations of this approach is that there is no fixed range in metrics to analyze that which one is less or which one is more complex. So, research is being carried out on different BPEL Process to analyze the range of metrics value.

The quality model, for assessment of high level design quality attributes have been developed and validated using different real world used web services. Soft computing technique is applied for designing the various models. Fuzzy logic is used to design the ISO standard quality attributes, i.e, FURPS model. It provides comparatively better result, and it can easily analyze different versions of software by using the rule viewer. Quality parameter are compared for different business processes.

Bibliography

[1] Rao, J., & Su, X..: "A survey of automated web service composition methods". In Semantic Web Services and Web Process Composition, Springer Berlin Heidelberg, pp. 43-54, 2005.

[2] Mendling, J., & Hafner, M..: "From WS-CDL choreography to BPEL process orchestration". Journal of Enterprise Information Management, vol.21, No. 5, pp. 525-542, 2008.

[3] Hamadi, R., & Benatallah, B..: "A Petri Net-based Model for Web Service Composition". Proc. of the 14th Australasian Database Conference , Australia, pp.191-200, 2003.

[4] Cardoso, J..: "Complexity analysis of BPEL web processes". Software Process: Improvement and Practice, vol. 12, no. 1, pp. 35-49, 2007.

[5] Cardoso, J..: "Business process control-flow complexity: Metric, evaluation, and validation". International Journal of Web Services Research, vol. 5, no. 2, pp. 49-76, 2008.

[6] Van der Aalst, W. M..: "The application of Petri nets to workflow management". Journal of circuits, systems, and computers, vol. 8, no. 01, pp. 21-66, 1998.

[7] Gyimothy, T., Ferenc, R., & Siket, I..: "Empirical validation of object-oriented metrics on open source software for fault prediction". Software Engineering, IEEE Transactions, vol. 31, no. 10, pp. 897-910, 2005.

[8] Gruhn, V., & Laue, R..: "Complexity metrics for business process models". In 9th international conference on business information systems, vol. 85, pp. 1-12, 2006.

[9] Bandi, Rajendra K., Vijay K. Vaishnavi, and Daniel E. Turk..: "Predicting maintenance performance using object-oriented design complexity metrics". Software Engineering, IEEE Transactions, vol. 29, no. 1, pp. 77-87, 2003.

[10] Cardoso, J..: "Process control-flow complexity metric: An empirical validation". In Services Computing SCC. IEEE International Conference, pp. 167-173, 2006.

[11] Menasc, Daniel A., Virgilio AF Almeida, and Larry W. Dowdy..: "Capacity Planning for Web Services: metrics, models, and methods". Upper Saddle River: Prentice Hall PTR, 2002.

[12] Jing, S., and Yu-yue, D..: “Performance analysis of web service composition based on stochastic well-formed workflow”. In Networked Computing (INC), pp. 1-5, 2010.

[13] Juan, Song, and Wang Hao. “Performance analysis for web service composition based on Queueing Petri Net”. In Software Engineering and Service Science (ICSESS), 2012 IEEE 3rd International Conference on, pp. 501-504. IEEE, 2012.

[14] Chang, H., Song, H., Kim, W., Lee, K., Park, H., Kwon, S., & Park, J. (2005). “Simulation-based web service composition: Framework and performance analysis”. In Systems Modeling and Simulation: Theory and Applications, pp. 352-360, 2005,

[15] Erickson, J., and Siau, K.: “Theoretical and practical complexity of modeling methods”. Communications of the ACM, vol. 50, no. 8, pp. 46-51, 2007.

[16] Hertis, M., & Juric, M..: “An Empirical Analysis of Business Process Execution Language Usage”. 2014.

[17] Hertis M. and Juric M. B..: “An empirical analysis of BPEL usage”. [Results], 2013. [Online]. Available: http://www.cloud.si/ BPELAnalysis/

[18] Landauer, T. K..: “The trouble with computers: Usefulness, usability, and productivity”. Cambridge, MA: MIT press, 1995.

[19] Liu, Y., Ngu, A. H., & Zeng, L. Z..: “QoS computation and policing in dynamic web service selection”. In Proceedings of the 13th international World Wide Web conference on Alternate track papers and posters, pp. 66-73, 2004.

[20] Zhang, J., & Zhang, L. J..: “Criteria analysis and validation of the reliability of web services-oriented systems”. 2005.

[21] Coleman, D., Ash, D., Lowther, B., & Oman, P..: “Using metrics to evaluate software system maintainability”. Computer, vol. 27, no. 8, pp. 44-49, 1994.

[22] Wohlin, C., Hst, M., Runeson, P., & Wessln, A..: “Software Reliability”. Encyclopedia of Physical Sciences and Technology, vol. 15, pp. 25-39, 2001.

[23] Suresh, Y., Kumar, L.,and Rath, S. K.: “Statistical and Machine Learning Methods for Software Fault Prediction Using CK Metric Suite: A Comparative Analysis”. International Scholarly Research Notices, vol. 2014, 2014

[24] Mamdani, E. H., and Assilian, S.: “An experiment in linguistic synthesis with a fuzzy logic controller”. International journal of man-machine studies, 7, (1), pp. 1-13

Dissemination

Journal

1. Debendra Kumar Naik and Santanu Kumar Rath, "Quality Assessment of web service composition using business process complexity metrics", *IET Software*, 2015 (Revised)

Conference

1. Naik, D. K., Kumari, S., & Rath, S. K.. "Application of Soft Computing Technique for Web Service Selection". In Distributed Computing and Internet Technology (pp. 245-248). Springer International Publishing.
2. Debendra Kumar Naik and Santanu Kumar Rath. "Risk Analysis of web service composition using design complexity metrics", *FICTA*, 2015 (Revised)
3. Lov Kumar, Debendra Kumar Naik and Santanu Kumar Rath, "Validating the Effectiveness of Object-Oriented Metrics for Predicting Maintainability", *ICRTC*, April 4-6, 2015 (Accepted)