

WSDL Specification of Services for Service Oriented Architecture (SOA) Based Implementation of A CRM Process

Prasenjit Kundu, Debabrata Das, Bikram Kesari Rath

Abstract— Recent research on SOA has opened new avenues for implementing various business processes through SOA architecture and CRM is no such exception. CRM is an important process of any business entity and CRM software implementation can be done through different available technology and methods. Recent research on CRM implementation has resulted the development of a conceptual model which can represent different sub processes within a CRM process as services under the SOA architecture but to implement this conceptual model in practice lots of consideration is required from technology and management point of view. WSDL is a XML grammar which can be used for SOA based services implementation in general and services description in particular. In this paper a conceptual model on SOA implementation of a CRM process has been the centre of discussion and the services identified by the conceptual model can be specified and described by WSDL which has been done actually in this paper using WSDL codes. The specification of services done in this paper can help the future researchers to actually implement the CRM process through SOA

Index Terms— CRM Software, SOA, Web Services, WSDL Specifications.

1 INTRODUCTION

SERVICE oriented architecture can be defined as a framework to integrate business processes and supporting IT infrastructures into secure, standardized components services that can be reused and combined to address changing business activities and priorities[1]. Different organizations across different sectors are adopting SOA just because of its capability to ensure business process improvement and thus in this way to gain competitive advantage[2]. The reason behind strong adaptation of SOA is its link between IT and business. Through SOA, services which require human interaction can be configured easily and systematically so that the whole system operates as per the customer needs. For this reason SOA reference architecture has become a clear framework for any enterprise operation [3]. Newcomer and Lomow believe that SOA should be seen as a style of design that provides guidance of creating and using business services throughout their lifecycle as well as defines and makes provisions for the IT infrastructure that allows different applications to exchange data and participate in business processes seamlessly irrespective of the operating systems or programming languages underlying those applications. The elementary SOA ingredients are (1) Infrastructure, (2) Architecture (3) Process and (4) Governance[4]. Business processes like CRM can be implemented under the SOA framework and for that first it is required to describe business processes as services and then services can be defined and specified by an extended form of XML grammar known as WSDL. In this paper an attempt has been made to implement a certain group of business processes as services using WSDL specifications.

2 LITERATURE REVIEW

PREMASA model proposes total eight steps. Some of these steps have only one activity & some steps have several activities. Out of these ten activities some are needed to be performed simultaneously & some are needed to be performed sequentially. The model is multi dimensional in nature because some of the activities are simultaneous & some of these activities are sequential. The model integrates these sequential & simultaneous activities into eight steps. The steps 4 & 5 have two simultaneous activities each and activities A4 & A4' as well as the activities A5 and A5' are needed to be performed simultaneously. All these ten activities have been arranged into five categories of phases & these are as follows:-

Activity A1 & A2 fall into Preparation Phase.

Activity A3, A4, & A4' fall into Measurement Phase.

Activity A5 & A5' fall into Action Phase.

Activity A6 falls into Satisfaction Phase.

Activity A7 & A8 fall into Application Phase.

The categorization of the activities in five phases has been done on the basis of the tasks performed by the activities. Another important basis of categorization of activities into five unique phases is that activities are grouped together as per their time of execution. Each phase denotes a unique period of time when the activity(ies) within that phase are needed to be performed [5]. Fig.1 shows the interrelationship of the activities, steps and phases of the PREMASA model.

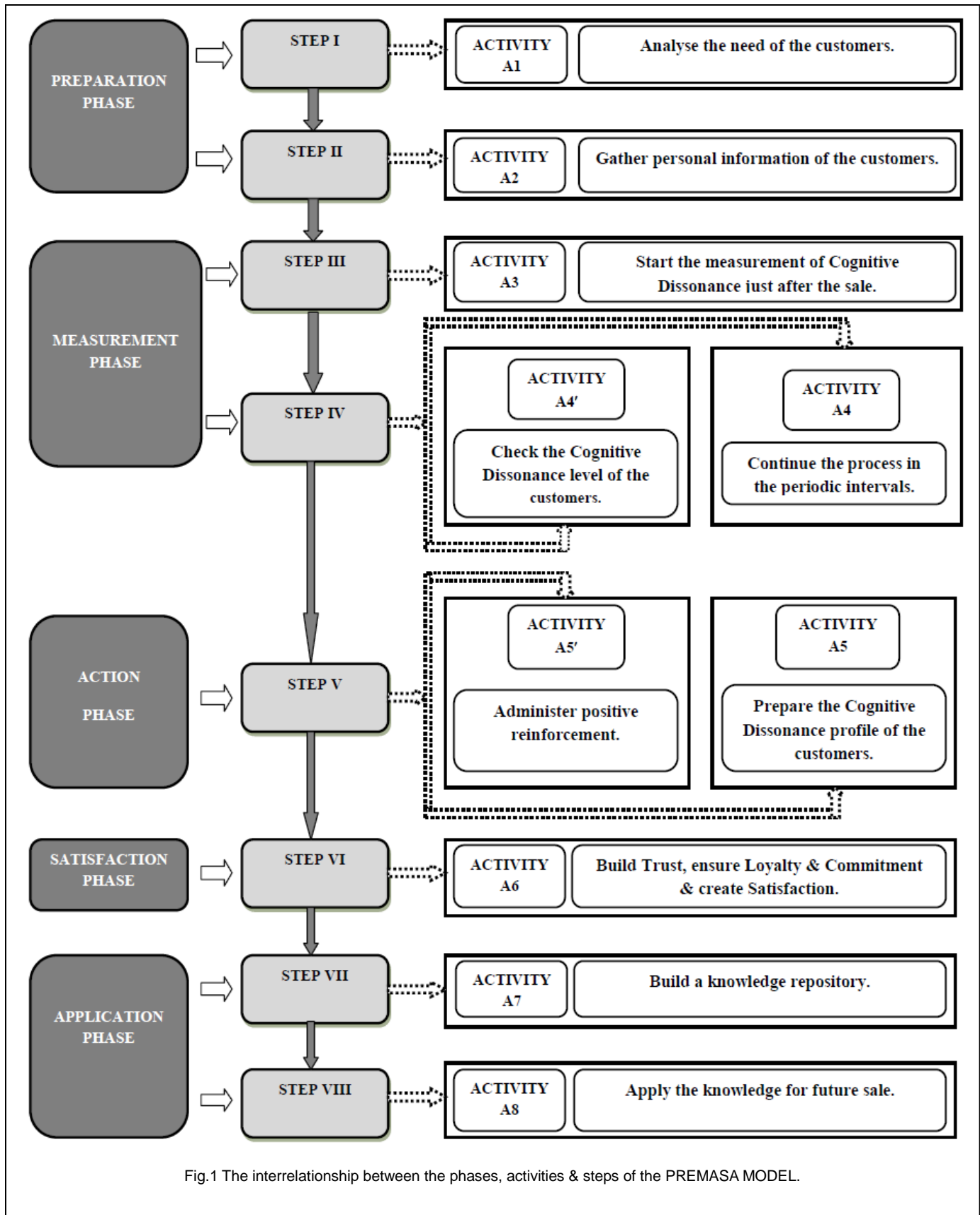


Fig.1 The interrelationship between the phases, activities & steps of the PREMASA MODEL.

In traditional CRM software there is no option to measure the cognitive dissonance and satisfaction levels of the customers and that is why the activities mentioned in the measurement and action phase of the PREMASA model need to be integrated and arrangements must be made so that these activities can be done through a CRM software. According to Krafzig, Banke and Slama, SOA is based on four key abstractions: (1) Application Front Ends (2) Service (3) Service Repository and (4) Service Bus or Enterprise Service Bus (ESB). Actually the Application Frontend is the owner of the business process while the services provide business functionality which the application frontends and other services can use. A service consists of an implementation that contains business logic and data, a service contract and a service interface. The service contract specifies the functionality, usage and constraints for a client of the service and a service interface physically exposes the functionality. The service repository stores the service contracts of the individual services of an SOA and the service bus (ESB) provide the linkage between the application front ends and services. A client can either be an application front end or service. It is always an application front end that initiates a business process and receives the result. Service is a software component of unique functional meaning that typically encapsulates a high level business concept. The application frontend basically interacts with the human user and that's why it is called sometimes the client. In some cases the client can be the service itself. The client uses the interfaces to invoke the service through the ESB (enterprise Service Bus). The interface is based on the service contract and the service contract describes the service and the service fulfills the service contracts and the service contract is stored in service repository [6].

Das and Kundu developed a conceptual model for implementing PREMASA Model through SOA approach [7]. This conceptual model is based on the approach adopted by Krafzig, Banke and Slama [6]. Das and Kundu tried to implement PREMASA Model from system level point of view and that's why they recommended that the first three phases of the PREMASA Model are the target phases which are needed to be streamlined and implemented as part of the CRM software. According to them the activities of these three phases can be transformed and grouped into five different services [7].

Researchers faced difficulty in using Web Services effectively because there was no mechanism to describe the web services. In response to the urgent need of describing various characteristics of a web service unambiguously WSDL was created [8]. WSDL or Web Services Definition Language can be defined as a specification for describing the methods or data types of SOAP interface [9]. WSDL is practically an XML format that describes the network services and its access information. It has a binding mechanism which is used to attach a protocol, data format, an abstract message or set of endpoints defining the location of services [10]. According to Ethan Cerami, WSDL specification defines how to describe web services in a common XML grammar. WSDL describes the following four pieces of data:

- Interface information that describes all publicly available functions.

- Data type information for all message requests and message responses.
- Binding information about the transport protocol which is needed to be used.
- Address information for locating a specified service.

In brief, WSDL represents a contract between the service requestor and the service provider. The most unique fact is that the WSDL is platform- and language-independent and is used primarily to describe SOAP services. Through WSDL, a client can locate a web service and invoke any of its publicly available functions. By using WSDL-aware tools, it is possible to automate this process, enabling applications to easily integrate new services with little or no manual code. That's why WSDL is the cornerstone of the web service architecture, because it provides a common language for describing services and a platform for automatically integrating those services. WSDL specification is an XML grammar for describing web services and the specification itself is divided into six major elements:

[1].Definitions

The first element is the definitions which is the root element of all WSDL documents. It defines the web service name and it declares multiple namespaces and it holds all the service elements.

[2].Types

The types element denotes all the data types used between the client and server. WSDL is not tied exclusively to a specific typing system, but by default it uses the W3C XML Schema specification. If the service uses only XML Schema built-in types, such as strings and integers, the types element is not required.

[3].Message

The message element describes a one-way message irrespective of whether it is a single message request or a single message response. It specifies the name of the message and contains zero or more message part elements, which can refer to message parameters or message return values.

[4].PortType

The fourth element is the PortType element which combines multiple message elements to form a complete one-way or round-trip operation. A portType can combine one request and one response message into a single request/response operation, most commonly used in SOAP services. In some cases a portType can define multiple operations.

[5].Binding

The fifth element is the binding element which describes the concrete specifics of how the service will be implemented on the wire.

[6].Service

The service element denotes the address for invoking the specified service. In general this includes a URL for invoking the SOAP service. In addition to the six major elements, the WSDL specification also has the following utility elements:

[a].Documentation

This element is used to provide human-readable documentation and can be included inside any other WSDL element.

[b].Import

The second utility element is used to import other WSDL documents or XML Schemas which enables more modular WSDL documents. Two WSDL documents can import the same basic elements and yet include their own service elements to make the same service available at two physical addresses but not all the WSDL tools support the import functionality as of yet[11].Fig 4 denotes a concise representation of the WSDL specification.

3 WSDL IMPLEMENTATION OF SOA SERVICES

Now taking cue from the Das and Kundu's approach in the paper "Application of Service Oriented Architecture (SOA) in Implementing a CRM Process: A Conceptual Approach", the five services can be further described and detailed using WSDL specification [7]. The way of specifying services using WSDL is based on the methods described by Ethan Cerami in the book "Web Services Essentials: Distributed Applications with XML-RPC, SOAP, and UDDI & WSDL" [11].

3.1 Service I

This service has been named as AnalyzeNeed_Service for the purpose of WSDL specification. In this service, the service contract is to analyze the need of the customers. In implementation, the business logic is to encode market research data regarding the need and demand of the customers. The data stores the encoder logic and the execution codes. This service provides an interface that supports two operations i.e. (1) Operation I: Calculate past demand and (2) Forecast Future Demand. Now for the first operation, there will be three message requests viz Ask_Past_Demand_M, Ask_Past_Demand_Y and Ask_Past_Demand_Q and three message responses viz Say_Past_Demand_M, Say_Past_Demand_Y and Say_Past_Demand_Q. For the three message requests the part names will be NameMonth, NameYear and NameQuarter and the types will be string. In the same manner there will be three binding names AskPD_M_Binding, AskPD_Y_Binding and AskPD_Q_Binding and three operation names AskPD_M, AskPD_Y and AskPD_Q. In the same manner there will be three port types and these are AskPD_M_PorTType, AskPD_Y_PorTType and AskPD_Q_PorTType. For the three port bindings the three names should be AskPD_M_PorT, AskPD_Q_PorT and AskPD_Y_PorT. (Refer Fig.2,3,4).

3.2 Service II

IJSER This service has been named as GatherPersonalInfo_Service for the purpose of WSDL specification. In service II, the service contract is to gather personal information of the customers. In implementation, the business logic is to encode customer info and segment & tag customers in terms of demographic variables. The data stores the encoder logic and the execution codes. Service II provides an interface that supports one operation i.e. (1) Operation I: Retrieve customers' data in

terms of demographic variables. Now for this operation, there will be four message requests viz Ask_Demo_Var_1, Ask_Demo_Var_2, Ask_Demo_Var_3 and Ask_Demo_Var_4 and the four message responses are Say_Cl_InV_1, Say_Cl_InV_2, Say_Cl_InV_3, and Say_Cl_InV_4. For the four message requests the part names will be Demo_Var_1, Demo_Var_2, Demo_Var_3 and Demo_Var_4 and the types will be string. In the same manner there will be four binding names Ask_DV1_Binding, Ask_DV2_Binding, Ask_DV3_Binding, Ask_DV4_Binding and the four operation names will be Ask_DV_1, Ask_DV_2, Ask_DV_3 and Ask_DV_4. In the same manner there will be four port types and these are Ask_DVPortType1, Ask_DVPortType2, Ask_DVPortType3 and Ask_DVPortType4. For the four port bindings the names should be Ask_DV1_PorT, Ask_DV2_PorT, Ask_DV3_PorT and Ask_DV4_PorT. (Refer Fig.5, 6, 7).

3.3 Service III

This service has been named as MeasurementCD_Service.wsdl for the purpose of WSDL specification. In service III, the service contract is to start the measurement of the cognitive dissonance (CD) of the customers just after the sale. In implementation, the business logic is to calculate & measure CD and encode the data. The data stores the encoder logic and the execution codes. Service III provides an interface that supports one operation i.e. (1) Operation I: Retrieve CD data of the customers in terms of demographic variables. Now for this operation, there will be four message requests viz Ask_CD_Var_1, Ask_CD_Var_2, Ask_CD_Var_3 and Ask_CD_Var_4 and the four message responses are Say_CD_InV_1, Say_CD_InV_2, Say_CD_InV_3 and Say_CD_InV_4. For the four message requests the part names will be CD_Var_1, CD_Var_2, CD_Var_3 and CD_Var_4 and the types will be string. For the four message response, the part names will be CD_InV_1, CD_InV_2, CD_InV_3 and CD_InV_4 and the types will be float. In the same manner there will be four binding names Ask_CD_V1_Binding, Ask_CD_V2_Binding, Ask_CD_V3_Binding and Ask_CD_V4_Binding and the four operation names Ask_CD_V_1, Ask_CD_V_2, Ask_CD_V_3 and Ask_CD_V_4. In the same manner there will be four port types and these are Ask_CDPorTType1, Ask_CDPorTType2, Ask_CDPorTType3 and Ask_CDPorTType4. For the four port bindings the four names should be Ask_CD_V1_PorT, Ask_CD_V2_PorT, Ask_CD_V3_PorT and Ask_CD_V4_PorT. (Refer Fig.8, 9, 10).

3.4 Service IV

This service has been named as CheckCDLevel_Service.wsdl for the purpose of WSDL specification. In service IV, the service contract is to check the CD level of the customers and continue the process in the periodic intervals. In implementation, the business logic is to check and compare the CD level of the customers with the predefined threshold CD level limits. The data stores the encoder logic and the execution codes. Service IV provides an interface that supports two operations i.e. (1) Operation I: Retrieve CD level of the customers in different time intervals and (2) Operation II: Compare and contrast the CD level of the customers across demographic variables. Now for this operation, there will be four message requests viz Ask_CD_tm_2, Ask_CD_tm_2,

Ask_CD_tm_3 and Ask_CD_tm_4 and the four message responses are Say_CDI_tm_1, Say_CDI_tm_2, Say_CDI_tm_3 and

WSDL Specification for Service I

AnalyzeNeed_Service.wsdl

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="AnalyzeNeed_Service"
  targetNamespace=http://www.premasa-model.com/ver1/wsdl/ AnalyzeNeed_Service.wsdl
  xmlns:impl="http://www.premasa-model.com/ver1/wsdl/ AnalyzeNeed_Service.wsdl "
  xmlns:intf="http://www.premasa-model.com/ver1/wsdl/ AnalyzeNeed_Service.wsdl "
  xmlns:tns="http://www.premasa-model.com/ver1/wsdl/ AnalyzeNeed_Service.wsdl "
```

```
  xmlns="http://schemas.xmlsoap.org/wsdl/"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:ws="http://ws-i.org/profiles/basic/1.1/xsd"
```

```
<message name="Ask_Past_Demand_M">
  <part name="NameMonth" type="xsd:string"/>
</message>
<message name="Say_Past_Demand_M">
  <part name="FigureValueMonth" type="xsd:float"/>
</message>
<message name="Ask_Past_Demand_Y">
  <part name="NameYear" type="xsd:string"/>
</message>
<message name="Say_Past_Demand_Y">
  <part name="FigureValueYear" type="xsd:float"/>
</message>
<message name="Ask_Past_Demand_Q">
  <part name="NameQuarter" type="xsd:string"/>
</message>
<message name="Say_Past_Demand_Q">
  <part name="FigureValueQuarter" type="xsd:float"/>
</message>

<portType name="AskPD_M_PorTType">
  <operation name="AskPD_M">
    <input message="tns:Ask_Past_Demand_M"/>
    <output message="tns:Say_Past_Demand_M"/>
  </operation>
</portType>
<portType name="AskPD_Y_PorTType">
  <operation name="AskPD_Y">
    <input message="tns:Ask_Past_Demand_Y"/>
    <output message="tns:Say_Past_Demand_Y"/>
  </operation>
</portType>
<portType name="AskPD_Q_PorTType">
  <operation name="AskPD_Q">
    <input message="tns:Ask_Past_Demand_Q"/>
    <output message="tns:Say_Past_Demand_Q"/>
  </operation>
</portType>
```

LEGEND

TEXT WITHIN THE BOX HIGHLIGHTED WITH PINK COLOUR -> Compulsory Code Elements






BLACK COLOURED TEXT		-> WSDL definitions components
BLUE COLOURED TEXT		-> WSDL message components
ORANGE COLOURED TEXT		-> WSDL porttype components
GREEN COLOURED TEXT		-> WSDL binding components
GOLD COLOURED TEXT		-> WSDL service components

Fig. 2.WSDL Specification for Service I

WSDL Specification for Service I

```
<binding name="AskPD_M_Binding" type="tns: AskPD_M_PorTType">
  <soap: binding style="rpc"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name=" AskPD_M ">
    <soap: operation soapAction=" AskPD_M "/>
    <input>
      <soap: body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: AnalyzeNeed_Service "
        use="encoded"/>
      </input>
      <output>
        <soap:body
          encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
          namespace="urn: examples: AnalyzeNeed_Service "
          use="encoded"/>
        </output>
      </operation>
    </binding>

<binding name="AskPD_Y_Binding" type="tns: AskPD_Y_PorTType">
  <soap: binding style="rpc"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name=" AskPD_Y ">
    <soap: operation soapAction=" AskPD_Y "/>
    <input>
      <soap: body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: AnalyzeNeed_Service "
        use="encoded"/>
      </input>
      <output>
        <soap:body
          encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
          namespace="urn: examples: AnalyzeNeed_Service "
          use="encoded"/>
        </output>
      </operation>
    </binding>
```

LEGEND

TEXT WITHIN THE BOX HIGHLIGHTED WITH PINK COLOUR -> Compulsory Code Elements

BLACK COLOURED TEXT		-> WSDL definations components
BLUE COLOURED TEXT		-> WSDL message components
ORANGE COLOURED TEXT		-> WSDL porttype components
GREEN COLOURED TEXT		-> WSDL binding components
GOLD COLOURED TEXT		-> WSDL service components

Fig. 3.WSDL Specification for Service I

WSDL Specification for Service I

```

<binding name="AskPD_Q_Binding" type="tns: AskPD_Q_PorTType">
  <soap: binding style="rpc"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name=" AskPD_Q ">
    <soap: operation soapAction=" AskPD_Q "/>
    <input>
      <soap: body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: AnalyzeNeed_Service "
        use="encoded"/>
    </input>
    <output>
      <soap:body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: AnalyzeNeed_Service "
        use="encoded" />
    </output>
  </operation>
</binding>

<service name=" AnalyzeNeed_Service ">
  <documentation>WSDL File for AnalyzeNeed_Service </documentation>
  <port binding="tns:AskPD_M_Binding" name="AskPD_M_PorT">
  <port binding="tns:AskPD_Y_Binding" name="AskPD_Y_PorT">
  <port binding="tns:AskPD_Q_Binding" name="AskPD_Q_PorT">
    <soap: address location="http://localhost:8080/soap/servlet/rpcrouter"/>
  </port>
</service>
</definitions>

```

LEGEND

TEXT WITHIN THE BOX HIGHLIGHTED WITH PINK COLOUR -> Compulsory Code Elements






BLACK COLOURED TEXT		-> WSDL definations components
BLUE COLOURED TEXT		-> WSDL message components
ORANGE COLOURED TEXT		-> WSDL porttype components
GREEN COLOURED TEXT		-> WSDL binding components
GOLD COLOURED TEXT		-> WSDL service components

Fig. 4.WSDL Specification for Service I

WSDL Specification for Service II

```
GatherPersonalInfo_Service.wsdl
<?xml version="1.0" encoding="UTF-8"?>
<definitions name=" GatherPersonalInfo_Service"
  targetNamespace=http://www.premasa-model.com/ver1/wsdl/ GatherPersonalInfo_Service.wsdl
  xmlns:impl=" http://www.premasa-model.com/ver1/wsdl/ GatherPersonalInfo_Service.wsdl "
  xmlns:intf=" http://www.premasa-model.com/ver1/wsdl/ GatherPersonalInfo_Service.wsdl "
  xmlns:tns=" http://www.premasa-model.com/ver1/wsdl/ GatherPersonalInfo_Service.wsdl "
```

```
  xmlns="http://schemas.xmlsoap.org/wsdl/"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  xmlns:wsdl="http://ws-i.org/profiles/basic/1.1/xsd"
```

```
<message name="Ask_Demo_Var_1">
  <part name=" Demo_Var_1" type="xsd: string"/>
</message>
<message name="Say_Cl_InV_1 ">
  <part name=" Cl_InV_1 " type="xsd: string "/>
</message>
<message name=" Ask_Demo_Var_2">
  <part name=" Demo_Var_2" type="xsd:string"/>
</message>
<message name=" Say_Cl_InV_2 ">
  <part name=" Cl_InV_2 " type="xsd:string"/>
</message>
<message name=" Ask_Demo_Var_3">
  <part name=" Demo_Var_3" type="xsd: string"/>
</message>
<message name=" Say_Cl_InV_3 ">
  <part name=" Cl_InV_3 " type="xsd:string"/>
</message>
<message name=" Ask_Demo_Var_4">
  <part name=" Demo_Var_4" type="xsd: string"/>
</message>
<message name=" Say_Cl_InV_4 ">
  <part name=" Cl_InV_4 " type="xsd:string"/>
</message>

<portType name=" Ask_DVPortType1 ">
  <operation name=" Ask_DV_1 ">
    <input message="tns: Ask_Demo_Var_1 "/>
    <output message="tns: Say_Cl_InV_1 "/>
  </operation>
</portType>

<portType name=" Ask_DVPortType2 ">
  <operation name=" Ask_DV_2 ">
    <input message="tns: Ask_Demo_Var_2 "/>
    <output message="tns: Say_Cl_InV_2 "/>
  </operation>
</portType>
```

LEGEND

TEXT WITHIN THE BOX HIGHLIGHTED WITH PINK COLOUR -> Compulsory Code Elements






BLACK COLOURED TEXT		-> WSDL definations components
BLUE COLOURED TEXT		-> WSDL message components
ORANGE COLOURED TEXT		-> WSDL porttype components
GREEN COLOURED TEXT		-> WSDL binding components
GOLD COLOURED TEXT		-> WSDL service components

Fig. 5.WSDL Specification for Service II

WSDL Specification for Service II

```

<portType name=" Ask_DVPortType3 ">
  <operation name=" Ask_DV_3 ">
    <input message="tns: Ask_Demo_Var_3 "/>
    <output message="tns: Say_Cl_InV_3 "/>
  </operation>
</portType>

<portType name=" Ask_DVPortType4 ">
  <operation name=" Ask_DV_4 ">
    <input message="tns: Ask_Demo_Var_4 "/>
    <output message="tns: Say_Cl_InV_4 "/>
  </operation>
</portType>

<binding name="Ask_DV1_Binding" type="tns: Ask_DVPortType1">
  <soap: binding style="rpc"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name=" Ask_DV_1 ">
    <soap: operation soapAction=" Ask_DV_1 "/>
    <input>
      <soap: body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: GatherPersonalInfo_Service "
        use="encoded"/>
    </input>
    <output>
      <soap:body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: GatherPersonalInfo_Service "
        use="encoded"/>
    </output>
  </operation>
</binding>

<binding name="Ask_DV2_Binding" type="tns: Ask_DVPortType2">
  <soap: binding style="rpc"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name=" Ask_DV_2 ">
    <soap: operation soapAction=" Ask_DV_2 "/>
    <input>
      <soap: body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: GatherPersonalInfo_Service "
        use="encoded"/>
    </input>
    <output>
      <soap:body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: GatherPersonalInfo_Service "
        use="encoded"/>
    </output>
  </operation>
</binding>

```

LEGEND






TEXT WITHIN THE BOX HIGHLIGHTED WITH PINK COLOUR -> Compulsory Code Elements		
BLACK COLOURED TEXT		-> WSDL definations components
BLUE COLOURED TEXT		-> WSDL message components
ORANGE COLOURED TEXT		-> WSDL porttype components
GREEN COLOURED TEXT		-> WSDL binding components
GOLD COLOURED TEXT		-> WSDL service components

Fig. 6.WSDL Specification for Service II

WSDL Specification for Service II

```

<binding name="Ask_DV3_Binding" type="tns: Ask_DVPortType3">
  <soap: binding style="rpc"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name=" Ask_DV_3 ">
    <soap: operation soapAction=" Ask_DV_3 "/>
    <input>
      <soap: body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: GatherPersonalInfo_Service "
        use="encoded"/>
    </input>
    <output>
      <soap:body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: GatherPersonalInfo_Service "
        use="encoded"/>
    </output>
  </operation>
</binding>

<binding name="Ask_DV4_Binding" type="tns: Ask_DVPortType4">
  <soap: binding style="rpc"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name=" Ask_DV_4 ">
    <soap: operation soapAction=" Ask_DV_4 "/>
    <input>
      <soap: body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: GatherPersonalInfo_Service "
        use="encoded"/>
    </input>
    <output>
      <soap:body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: GatherPersonalInfo_Service "
        use="encoded"/>
    </output>
  </operation>
</binding>

</definitions>
<service name=" GatherPersonalInfo_Service ">
  <documentation>WSDL File for GatherPersonalInfo_Service </documentation>
  <port binding="tns: Ask_DV1_Binding " name=" Ask_DV1_PorT " >
  <port binding="tns: Ask_DV2_Binding " name=" Ask_DV2_PorT " >
  <port binding="tns: Ask_DV3_Binding " name=" Ask_DV3_PorT " >
  <port binding="tns: Ask_DV4_Binding " name=" Ask_DV4_PorT " >
  <soap: address location="http://localhost:8080/soap/servlet/rpcrouter"/>
</port>
</service>
</definitions>

```

LEGEND

TEXT WITHIN THE BOX HIGHLIGHTED WITH PINK COLOUR -> Compulsory Code Elements




BLACK COLOURED TEXT		-> WSDL definations components
BLUE COLOURED TEXT		-> WSDL message components
ORANGE COLOURED TEXT		-> WSDL porttype components
GREEN COLOURED TEXT		-> WSDL binding components
GOLD COLOURED TEXT		-> WSDL service components

Fig. 7.WSDL Specification for Service II

WSDL Specification for Service III

MeasurementCD_Service.wsdl

```
<? xml version="1.0" encoding="UTF-8"?>
<definitions name=" MeasurementCD_Service"
  targetNamespace=http://www.premasa-model.com/ver1/wsdl/ MeasurementCD_Service.wsdl
  xmlns:impl=" http://www.premasa-model.com/ver1/wsdl/ MeasurementCD_Service.wsdl "
  xmlns:intf=" http://www.premasa-model.com/ver1/wsdl/ MeasurementCD_Service.wsdl "
  xmlns:tns=" http://www.premasa-model.com/ver1/wsdl/ MeasurementCD_Service.wsdl "
```

```
  xmlns="http://schemas.xmlsoap.org/wsdl/"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:ws="http://ws-i.org/profiles/basic/1.1/xsd"
```

```
<message name="Ask_CD_Var_1">
  <part name=" CD_Var_1" type="xsd: string"/>
</message>
<message name="Say_CD_InV_1 ">
  <part name=" CD_InV_1 " type="xsd: float "/>
</message>
<message name="Ask_CD_Var_2">
  <part name=" CD_Var_2" type="xsd: string"/>
</message>
<message name="Say_CD_InV_2 ">
  <part name=" CD_InV_2 " type="xsd: float "/>
</message>
<message name="Ask_CD_Var_3">
  <part name=" CD_Var_3" type="xsd: string"/>
</message>
<message name="Say_CD_InV_3 ">
  <part name=" CD_InV_3 " type="xsd: float "/>
</message>
<message name="Ask_CD_Var_4">
  <part name=" CD_Var_4" type="xsd: string"/>
</message>
<message name="Say_CD_InV_4 ">
  <part name=" CD_InV_4 " type="xsd: float "/>
</message>
```

```
<portType name=" Ask_CDPortType1 ">
  <operation name=" Ask_CD_V_1 ">
    <input message="tns: Ask_CD_Var_1 "/>
    <output message="tns: Say_CD_InV_1 "/>
  </operation>
</portType>
<portType name=" Ask_CDPortType2 ">
  <operation name=" Ask_CD_V_2 ">
    <input message="tns: Ask_CD_Var_2 "/>
    <output message="tns: Say_CD_InV_2 "/>
  </operation>
</portType>
```

LEGEND

TEXT WITHIN THE BOX HIGHLIGHTED WITH PINK COLOUR -> Compulsory Code Elements

BLACK COLOURED TEXT		-> WSDL definations components
BLUE COLOURED TEXT		-> WSDL message components
ORANGE COLOURED TEXT		-> WSDL porttype components
GREEN COLOURED TEXT		-> WSDL binding components
GOLD COLOURED TEXT		-> WSDL service components

Fig. 8.WSDL Specification for Service III

WSDL Specification for Service III

```

<portType name=" Ask_CDPortType3 ">
  <operation name=" Ask_CD_V_3 ">
    <input message="tns: Ask_CD_Var_3 "/>
    <output message="tns: Say_CD_InV_3 "/>
  </operation>
</portType>
<portType name=" Ask_CDPortType4 ">
  <operation name=" Ask_CD_V_4 ">
    <input message="tns: Ask_CD_Var_4 "/>
    <output message="tns: Say_CD_InV_4 "/>
  </operation>
</portType>

<binding name="Ask_CD_V1_Binding" type="tns: Ask_CDPortType1">
  <soap: binding style="rpc"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name=" Ask_CD_V_1 ">
    <soap: operation soapAction=" Ask_CD_V_1 "/>
    <input>
      <soap: body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: MeasurementCD_Service "
        use="encoded"/>
    </input>
    <output>
      <soap:body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: MeasurementCD_Service "
        use="encoded"/>
    </output>
  </operation>
</binding>

<binding name="Ask_CD_V2_Binding" type="tns: Ask_CDPortType2">
  <soap: binding style="rpc"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name=" Ask_CD_V_2 ">
    <soap: operation soapAction=" Ask_CD_V_2 "/>
    <input>
      <soap: body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: MeasurementCD_Service "
        use="encoded"/>
    </input>
    <output>
      <soap:body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: MeasurementCD_Service "
        use="encoded"/>
    </output>
  </operation>
</binding>

```

LEGEND

TEXT WITHIN THE BOX HIGHLIGHTED WITH PINK COLOUR -> Compulsory Code Elements






BLACK COLOURED TEXT		-> WSDL definations components
BLUE COLOURED TEXT		-> WSDL message components
ORANGE COLOURED TEXT		-> WSDL porttype components
GREEN COLOURED TEXT		-> WSDL binding components
GOLD COLOURED TEXT		-> WSDL service components

Fig. 9.WSDL Specification for Service III

WSDL Specification for Service III

```

<binding name="Ask_CD_V3_Binding" type="tns: Ask_CDPorTType3">
  <soap: binding style="rpc"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name=" Ask_CD_V_3 ">
    <soap: operation soapAction=" Ask_CD_V_3 "/>
    <input>
      <soap: body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: MeasurementCD_Service "
        use="encoded"/>
    </input>
    <output>
      <soap:body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: MeasurementCD_Service "
        use="encoded"/>
    </output>
  </operation>
</binding>

<binding name="Ask_CD_V4_Binding" type="tns: Ask_CDPorTType4">
  <soap: binding style="rpc"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name=" Ask_CD_V_4 ">
    <soap: operation soapAction=" Ask_CD_V_4 "/>
    <input>
      <soap: body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: MeasurementCD_Service "
        use="encoded"/>
    </input>
    <output>
      <soap:body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: MeasurementCD_Service "
        use="encoded"/>
    </output>
  </operation>
</binding>

<service name=" MeasurementCD_Service ">
  <documentation>WSDL File for MeasurementCD_Service </documentation>
  <port binding="tns: Ask_CD_V1_Binding " name=" Ask_CD_V1_PorT " >
  <port binding="tns: Ask_CD_V2_Binding " name=" Ask_CD_V2_PorT " >
  <port binding="tns: Ask_CD_V3_Binding " name=" Ask_CD_V3_PorT " >
  <port binding="tns: Ask_CD_V4_Binding " name=" Ask_CD_V4_PorT " >
  <soap:address location="http://localhost:8080/soap/servlet/rpcrouter"/>
</port>
</service>
</definitions>

```

LEGEND

TEXT WITHIN THE BOX HIGHLIGHTED WITH PINK COLOUR -> Compulsory Code Elements






BLACK COLOURED TEXT		-> WSDL definations components
BLUE COLOURED TEXT		-> WSDL message components
ORANGE COLOURED TEXT		-> WSDL porttype components
GREEN COLOURED TEXT		-> WSDL binding components
GOLD COLOURED TEXT		-> WSDL service components

Fig. 10.WSDL Specification for Service III

WSDL Specification for Service IV

CheckCDLevel_Service.wsdl

```
<? xml version="1.0" encoding="UTF-8"?>
<definitions name=" CheckCDLevel_Service"
  targetNamespace=http://www.premasa-model.com/ver1/wsdl/ CheckCDLevel_Service.wsdl
  xmlns:impl =" http://www.premasa-model.com/ver1/wsdl/ CheckCDLevel_Service.wsdl "
  xmlns:intf =" http://www.premasa-model.com/ver1/wsdl/ CheckCDLevel_Service.wsdl "
  xmlns:tns=" http://www.premasa-model.com/ver1/wsdl/ CheckCDLevel_Service.wsdl "
```

```
xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
```

```
<message name="Ask_CD_tm_1">
  <part name=" CD_tm_1" type="xsd: string"/>
</message>
<message name="Say_CD_tm_1 ">
  <part name=" CDL_tm_1" type="xsd: string "/>
</message>
<message name="Ask_CD_tm_2">
  <part name=" CD_tm_2" type="xsd: string"/>
</message>
<message name="Say_CD_tm_2 ">
  <part name=" CDL_tm_2" type="xsd: string "/>
</message>
<message name="Ask_CD_tm_3">
  <part name=" CD_tm_3" type="xsd: string"/>
</message>
<message name="Say_CD_tm_3 ">
  <part name=" CDL_tm_3" type="xsd: string "/>
</message>
<message name="Ask_CD_tm_4">
  <part name=" CD_tm_4" type="xsd: string"/>
</message>
<message name="Say_CD_tm_4 ">
  <part name=" CDL_tm_4" type="xsd: string "/>
</message>
<message name="Ask_CD_tm_5">
  <part name=" CD_tm_5" type="xsd: string"/>
</message>
<message name="Say_CD_tm_5 ">
  <part name=" CDL_tm_5" type="xsd: string "/>
</message>

<portType name=" Ask_CD_PortType1 ">
  <operation name=" Ask_CD_tnv_1 ">
    <input message="tns: Ask_CD_tm_1 "/>
    <output message="tns: Say_CD_tm_1 "/>
  </operation>
</portType>
<portType name=" Ask_CD_PortType2 ">
  <operation name=" Ask_CD_tnv_2 ">
    <input message="tns: Ask_CD_tm_2 "/>
    <output message="tns: Say_CD_tm_2 "/>
  </operation>
</portType>
```

LEGEND

TEXT WITHIN THE BOX HIGHLIGHTED WITH PINK COLOUR -> Compulsory Code Elements

BLACK COLOURED TEXT		-> WSDL definitions components
BLUE COLOURED TEXT		-> WSDL message components
ORANGE COLOURED TEXT		-> WSDL porttype components
GREEN COLOURED TEXT		-> WSDL binding components
GOLD COLOURED TEXT		-> WSDL service components

Fig. 11. WSDL Specification for Service IV

WSDL Specification for Service IV

```

<portType name=" Ask_CDL_PortType3 ">
  <operation name=" Ask_CD_tnv_3 ">
    <input message="tns: Ask_CD_tm_3 "/>
    <output message="tns: Say_CDL_tm_3 "/>
  </operation>
</portType>
<portType name=" Ask_CDL_PortType4 ">
  <operation name=" Ask_CD_tnv_4 ">
    <input message="tns: Ask_CD_tm_4 "/>
    <output message="tns: Say_CDL_tm_4 "/>
  </operation>
</portType>
<portType name=" Ask_CDL_PortType5 ">
  <operation name=" Ask_CD_tnv_5 ">
    <input message="tns: Ask_CD_tm_5 "/>
    <output message="tns: Say_CDL_tm_5 "/>
  </operation>
</portType>

<binding name="Ask_CD_tm1_Binding" type="tns: Ask_CDL_PortType1">
  <soap: binding style="rpc"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name=" Ask_CD_tnv_1 ">
    <soap: operation soapAction=" Ask_CD_tnv_1 "/>
    <input>
      <soap: body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: CheckCDLevel_Service "
        use="encoded"/>
    </input>
    <output>
      <soap:body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: CheckCDLevel_Service.wsdll "
        use="encoded"/>
    </output>
  </operation>
</binding>
<binding name="Ask_CD_tm2_Binding" type="tns: Ask_CDL_PortType2">
  <soap: binding style="rpc"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name=" Ask_CD_tnv_2 ">
    <soap: operation soapAction=" Ask_CD_tnv_2 "/>
    <input>
      <soap: body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: CheckCDLevel_Service "
        use="encoded"/>
    </input>
    <output>
      <soap:body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: CheckCDLevel_Service.wsdll "
        use="encoded"/>
    </output>
  </operation>
</binding>

```

LEGEND

TEXT WITHIN THE BOX HIGHLIGHTED WITH PINK COLOUR -> Compulsory Code Elements

BLACK COLOURED TEXT		-> WSDL definitions components
BLUE COLOURED TEXT		-> WSDL message components
ORANGE COLOURED TEXT		-> WSDL porttype components
GREEN COLOURED TEXT		-> WSDL binding components
GOLD COLOURED TEXT		-> WSDL service components

Fig. 12 .WSDL Specification for Service IV

WSDL Specification for Service IV

```

<binding name="Ask_CD_tm3_Binding" type="tns: Ask_CD_PortType3">
  <soap: binding style="rpc"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name=" Ask_CD_tnv_3 ">
    <soap: operation soapAction=" Ask_CD_tnv_3 "/>
    <input>
      <soap: body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: CheckCDLevel_Service "
        use="encoded"/>
      </input>
      <output>
        <soap:body
          encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
          namespace="urn: examples: CheckCDLevel_Service.wsdl "
          use="encoded"/>
        </output>
      </operation>
    </binding>

<binding name="Ask_CD_tm4_Binding" type="tns: Ask_CD_PortType4">
  <soap: binding style="rpc"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name=" Ask_CD_tnv_4 ">
    <soap: operation soapAction=" Ask_CD_tnv_4 "/>
    <input>
      <soap: body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: CheckCDLevel_Service "
        use="encoded"/>
      </input>
      <output>
        <soap:body
          encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
          namespace="urn: examples: CheckCDLevel_Service.wsdl "
          use="encoded"/>
        </output>
      </operation>
    </binding>
  
```

LEGEND

TEXT WITHIN THE BOX HIGHLIGHTED WITH PINK COLOUR -> Compulsory Code Elements






BLACK COLOURED TEXT		-> WSDL definitions components
BLUE COLOURED TEXT		-> WSDL message components
ORANGE COLOURED TEXT		-> WSDL porttype components
GREEN COLOURED TEXT		-> WSDL binding components
GOLD COLOURED TEXT		-> WSDL service components

Fig. 13.WSDL Specification for Service IV






WSDL Specification for Service IV

```
<binding name="Ask_CD_tm5_Binding" type="tns: Ask_CDL_PortType5">
  <soap: binding style="rpc"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name=" Ask_CD_tnv_5 ">
    <soap: operation soapAction=" Ask_CD_tnv_5 "/>
    <input>
      <soap: body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: CheckCDLevel_Service "
        use="encoded"/>
    </input>
    <output>
      <soap:body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: CheckCDLevel_Service.wsdl "
        use="encoded"/>
    </output>
  </operation>
</binding>

<service name=" CheckCDLevel_Service ">
  <documentation>WSDL File for MeasurementCD_Service </documentation>
  <port binding="tns: Ask_CD_tm1_Binding " name=" Ask_CD_tm1_PorT " >
  <port binding="tns: Ask_CD_tm2_Binding " name=" Ask_CD_tm2_PorT " >
  <port binding="tns: Ask_CD_tm3_Binding " name=" Ask_CD_tm3_PorT " >
  <port binding="tns: Ask_CD_tm4_Binding " name=" Ask_CD_tm4_PorT " >
  <port binding="tns: Ask_CD_tm5_Binding " name=" Ask_CD_tm5_PorT " >
  <soap:address location="http://localhost:8080/soap/servlet/rpcrouter"/>
</port>
</service>
</definitions>
```

LEGEND

TEXT WITHIN THE BOX HIGHLIGHTED WITH PINK COLOUR -> Compulsory Code Elements

BLACK COLOURED TEXT		-> WSDL definations components
BLUE COLOURED TEXT		-> WSDL message components
ORANGE COLOURED TEXT		-> WSDL porttype components
GREEN COLOURED TEXT		-> WSDL binding components
GOLD COLOURED TEXT		-> WSDL service components

WSDL Specification for Service V

CreateCDProfile_Service.wsdl

```
<? xml version="1.0" encoding="UTF-8"?>
<definitions name=" CreateCDProfile_Service"
  targetNamespace=http://www.premasa-model.com/ver1/wsdl/ CreateCDProfile_Service.wsdl
  xmlns:impl =" http://www.premasa-model.com/ver1/wsdl/ CreateCDProfile_Service.wsdl "
  xmlns:intf =" http://www.premasa-model.com/ver1/wsdl/ CreateCDProfile_Service.wsdl "
  xmlns:tns=" http://www.premasa-model.com/ver1/wsdl/ CreateCDProfile_Service.wsdl "
```

```
xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:ws="http://ws-i.org/profiles/basic/1.1/xsd"
```

```
<message name="Ask_CDP_t_1">
  <part name=" CDP_t_1" type="xsd: string"/>
</message>
<message name="List_CDP_tm_1 ">
  <part name=" CDP_tm_1" type="xsd: string "/>
</message>
<message name="Ask_CDP_t_2">
  <part name=" CDP_t_2" type="xsd: string"/>
</message>
<message name="List_CDP_tm_2 ">
  <part name=" CDP_tm_2" type="xsd: string "/>
</message>
<message name="Ask_CDP_t_3">
  <part name=" CDP_t_3" type="xsd: string"/>
</message>
<message name="List_CDP_tm_3 ">
  <part name=" CDP_tm_3" type="xsd: string "/>
</message>
<message name="Ask_CDP_t_4">
  <part name=" CDP_t_4" type="xsd: string"/>
</message>
<message name="List_CDP_tm_4 ">
  <part name=" CDP_tm_4" type="xsd: string "/>
</message>
<message name="Ask_CDP_t_5">
  <part name=" CDP_t_5" type="xsd: string"/>
</message>
<message name="List_CDP_tm_5 ">
  <part name=" CDP_tm_5" type="xsd: string "/>
</message>
```

```
<portType name=" Ask_CDP_Portt_1 ">
  <operation name=" Ask_CDP_tm_1 ">
    <input message="tns: Ask_CDP_t_1 "/>
    <output message="tns: List_CDP_tm_1 "/>
  </operation>
</portType>
<portType name=" Ask_CDP_Portt_2 ">
  <operation name=" Ask_CDP_tm_2 ">
    <input message="tns: Ask_CDP_t_2 "/>
    <output message="tns: List_CDP_tm_2 "/>
  </operation>
</portType>
```

LEGEND

TEXT WITHIN THE BOX HIGHLIGHTED WITH PINK COLOUR -> Compulsory Code Elements

BLACK COLOURED TEXT		-> WSDL definitions components
BLUE COLOURED TEXT		-> WSDL message components
ORANGE COLOURED TEXT		-> WSDL porttype components
GREEN COLOURED TEXT		-> WSDL binding components
GOLD COLOURED TEXT		-> WSDL service components

WSDL Specification for Service V

```

<portType name=" Ask_CDP_Portt_3 ">
  <operation name=" Ask_CDP_tm_3 ">
    <input message="tns: Ask_CDP_t_3 "/>
    <output message="tns: List_CDP_tm_3 "/>
  </operation>
</portType>
<portType name=" Ask_CDP_Portt_4 ">
  <operation name=" Ask_CDP_tm_4 ">
    <input message="tns: Ask_CDP_t_4 "/>
    <output message="tns: List_CDP_tm_4 "/>
  </operation>
</portType>
<portType name=" Ask_CDP_Portt_5 ">
  <operation name=" Ask_CDP_tm_5 ">
    <input message="tns: Ask_CDP_t_5 "/>
    <output message="tns: List_CDP_tm_5 "/>
  </operation>
</portType>

<binding name="Ask_CDP_tm1_Binding" type="tns: Ask_CDP_Portt_1">
  <soap: binding style="rpc"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name=" Ask_CDP_tm_1 ">
    <soap: operation soapAction=" Ask_CDP_tm_1 "/>
    <input>
      <soap: body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: CreateCDProfile_Service.wsdl "
        use="encoded"/>
      </input>
      <output>
        <soap:body
          encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
          namespace="urn: examples: CreateCDProfile_Service.wsdl "
          use="encoded"/>
        </output>
      </operation>
    </binding>
  <binding name="Ask_CDP_tm2_Binding" type="tns: Ask_CDP_Portt_2">
    <soap: binding style="rpc"
      transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name=" Ask_CDP_tm_2 ">
      <soap: operation soapAction=" Ask_CDP_tm_2 "/>
      <input>
        <soap: body
          encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
          namespace="urn: examples: CreateCDProfile_Service.wsdl "
          use="encoded"/>
        </input>
        <output>
          <soap:body
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="urn: examples: CreateCDProfile_Service.wsdl "
            use="encoded"/>
          </output>
        </operation>
      </binding>
    
```

LEGEND

TEXT WITHIN THE BOX HIGHLIGHTED WITH PINK COLOUR -> Compulsory Code Elements

BLACK COLOURED TEXT		-> WSDL definitions components
BLUE COLOURED TEXT		-> WSDL message components
ORANGE COLOURED TEXT		-> WSDL porttype components
GREEN COLOURED TEXT		-> WSDL binding components
GOLD COLOURED TEXT		-> WSDL service components

Fig. 16.WSDL Specification for Service V

WSDL Specification for Service V

```

<binding name="Ask_CDP_tm3_Binding" type="tns: Ask_CDP_Portt_3">
  <soap: binding style="rpc"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name=" Ask_CDP_tm_3 ">
    <soap: operation soapAction=" Ask_CDP_tm_3 "/>
    <input>
      <soap: body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: CreateCDProfile_Service.wsdl "
        use="encoded"/>
      </input>
      <output>
        <soap:body
          encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
          namespace="urn: examples: CreateCDProfile_Service.wsdl "
          use="encoded"/>
        </output>
      </operation>
    </binding>

<binding name="Ask_CDP_tm4_Binding" type="tns: Ask_CDP_Portt_4">
  <soap: binding style="rpc"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name=" Ask_CDP_tm_4 ">
    <soap: operation soapAction=" Ask_CDP_tm_4 "/>
    <input>
      <soap: body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: CreateCDProfile_Service.wsdl "
        use="encoded"/>
      </input>
      <output>
        <soap:body
          encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
          namespace="urn: examples: CreateCDProfile_Service.wsdl "
          use="encoded"/>
        </output>
      </operation>
    </binding>
  
```

LEGEND

TEXT WITHIN THE BOX HIGHLIGHTED WITH PINK COLOUR -> Compulsory Code Elements






BLACK COLOURED TEXT		-> WSDL definitions components
BLUE COLOURED TEXT		-> WSDL message components
ORANGE COLOURED TEXT		-> WSDL porttype components
GREEN COLOURED TEXT		-> WSDL binding components
GOLD COLOURED TEXT		-> WSDL service components

Fig. 17 WSDL Specification for Service V

WSDL Specification for Service V

```

<binding name="Ask_CDP_tm5_Binding" type="tns: Ask_CDP_Portt_5">
  <soap: binding style="rpc"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name=" Ask_CDP_tm_5 ">
    <soap: operation soapAction=" Ask_CDP_tm_5 "/>
    <input>
      <soap: body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn: examples: CreateCDProfile_Service.wsdl "
        use="encoded"/>
      </input>
      <output>
        <soap:body
          encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
          namespace="urn: examples: CreateCDProfile_Service.wsdl "
          use="encoded"/>
        </output>
      </operation>
    </binding>

<service name=" CreateCDProfile_Service ">
  <documentation>WSDL File for MeasurementCD_Service </documentation>
  <port binding="tns: Ask_CDP_tm1_Binding " name=" Ask_CDP_tm1_PorT " >
  <port binding="tns: Ask_CDP_tm1_Binding " name=" Ask_CDP_tm1_PorT " >
  <port binding="tns: Ask_CDP_tm1_Binding " name=" Ask_CDP_tm1_PorT " >
  <port binding="tns: Ask_CDP_tm1_Binding " name=" Ask_CDP_tm1_PorT " >
  <port binding="tns: Ask_CDP_tm1_Binding " name=" Ask_CDP_tm1_PorT " >
  <soap: address location="http://localhost:8080/soap/servlet/rpcrouter"/>
</port>
</service>
</definitions>
  
```

LEGEND

TEXT WITHIN THE BOX HIGHLIGHTED WITH PINK COLOUR -> Compulsory Code Elements

BLACK COLOURED TEXT		-> WSDL definations components
BLUE COLOURED TEXT		-> WSDL message components
ORANGE COLOURED TEXT		-> WSDL porttype components
GREEN COLOURED TEXT		-> WSDL binding components
GOLD COLOURED TEXT		-> WSDL service components

Fig. 18.WSDL Specification for Service V

Say_CDL_tm_4. For the four message requests the part names will be CD_tm_1, CD_tm_2, CD_tm_3 and CD_tm_4 and the types will be string. For the four message response, the part names will be CDL_tm_1, CDL_tm_2, CDL_tm_3 and CDL_tm_4 and the types will be string. In the same manner there will be four binding names Ask_CD_tm1_Binding, Ask_CD_tm2_Binding, Ask_CD_tm3_Binding and Ask_CD_tm4_Binding and the four operation names Ask_CD_tnv_1, Ask_CD_tnv_2, Ask_CD_tnv_3 and Ask_CD_tnv_4. In the same manner there will be four port types and these are Ask_CDL_PortType1, Ask_CDL_PortType2, Ask_CDL_PortType3 and Ask_CDL_PortType4. For the four port bindings the four names should be Ask_CD_tm1_PorT, Ask_CD_tm2_PorT, Ask_CD_tm3_PorT and Ask_CD_tm4_PorT. (Refer Fig.11, 12, 13, 14).

3.5 Service V

This service has been named as CreateCDProfile_Service.wsdl for the purpose of WSDL specification. In service V, the service contract is to create the CD profile of the customers. In implementation, the business logic is to create and store CD profile of the customers using multiple tags where the tags denote time interval and demographic variables. The data stores the encoder logic and the execution codes. Service V provides an interface that supports one operation i.e. (1) Operation I: Retrieve & list CD profile of the customers across different time intervals. Again application front ends must be designed to give the user freedom of operation on the activities under the three phases of preparation, measurement and action. Now for this operation, there will be five message requests viz Ask_CDP_t_1, Ask_CDP_t_2, Ask_CDP_t_3, Ask_CDP_t_4 and Ask_CDP_t_5 and the five message responses are List_CDP_tm_1, List_CDP_tm_2, List_CDP_tm_3, List_CDP_tm_4 and List_CDP_tm_5. For the five message requests the part names will be CDP_t_1, CDP_t_2, CDP_t_3, CDP_t_4 and CDP_t_5 and the types will be string. For the five message responses, the part names will be CDP_tm_1, CDP_tm_2, CDP_tm_3, CDP_tm_4 and CDP_tm_5 and the types will be string. In the same manner there will be five binding names Ask_CDP_tm1_Binding, Ask_CDP_tm2_Binding, Ask_CDP_tm3_Binding, Ask_CDP_tm4_Binding and Ask_CDP_tm5_Binding and the five operation names Ask_CDP_tm_1, Ask_CDP_tm_2, Ask_CDP_tm_3, Ask_CDP_tm_4 and Ask_CDP_tm_5. In the same manner there will be five port types and these are Ask_CDP_Portt_1, Ask_CDP_Portt_2, Ask_CDP_Portt_3, Ask_CDP_Portt_4 and Ask_CDP_Portt_5. For the five port bindings the five names should be Ask_CDP_tm1_PorT, Ask_CDP_tm2_PorT, Ask_CDP_tm3_PorT, Ask_CDP_tm4_PorT and Ask_CDP_tm5_PorT. (Refer Fig.15, 16, 17, 18).

4 LIMITATIONS & FUTURE DIRECTIONS

The job of specifying the services through WSDL is more conceptual than technical in nature and that's why the description

and specification of services through WSDL is not universal in nature and can vary according to the need of the programmer as well as per the requirement specified by the clients and so this specification and description is not final and absolute and lots of rearrangement can be carried out in future. WSDL specification of services is the first step towards the SOA implementation of the conceptual CRM model and actual implementation requires next level of research which is designing the WSDL Invocation Tools as per the IBM Web Services Invocation Framework (WSIF) [11].

Dynamic program analysis is the analysis of computer software that is performed by executing programs built from that software system on a real time basis and it attempts to tune the application software during execution without stopping, recompiling or even rerunning the application. To achieve this objective it is required to use dynamic instrumentation techniques that allow the modification of the application code on the fly ([12], [13]). Program instrumentation is a general way to understand and analyze what an executing program is doing [14]. Program instrumentation eliminates the need to modify or recompile the application's source and it also supports the instrumentation of programs that dynamically generate code [15]. Kundu proposes that it is also possible to change instrumentation at any time during execution by modifying the application's binary image as well as the runtime behaviour can be extracted from the software using software visualization [16]. Based on this, the research on the conceptual analysis of the runtime behaviour of a CRM software by Kundu and Das reveals that it is possible to identify the pitfalls of a CRM software through dynamic program analysis and it is also possible to resolve the pitfalls through plug in instrumentation [17].

Recent research has also proved that the concept of plug-in instrumentation can be used to insert extra code in the existing CRM software package at runtime without altering its execution. Das and Kundu has proposed a plug in instrumentation technique through which the measurement module containing the measurement phase of the PREMASA model can be inserted in a CRM software package without altering or hampering the normal execution of the software at runtime [18].

Convergence of the two approaches and technologies (SOA and Dynamic Program Instrumentation) will open up infinite possibilities for conceptual and theoretical modeling and will also pave the way for developing an integrated and combined approach for business process reengineering in near future.

5 CONCLUSIONS

Despite several limitations arising out of the question of practicality in implementation, this paper gives a clear direction for future research on SOA implementation of the PREMASA model and also this paper will guide the researchers towards developing conceptual models of business process reengineering in future.

REFERENCES

- [1] N.Bieberstein, S.Bose, M. Fiammante, K.Jones and R.Shah, *Service Oriented Architecture Compass: Business Value, Planning and Enterprise Roadmap*. Upper Saddle River, NJ: Pearson Education, Inc, 2006.
- [2] J.P.Lawler, H.Howell- Barber, *Service Oriented Architecture: SOA Strategy, Methodology and Technology*. Auerbach Publications, 2007.
- [3] N.Bieberstein, R.G.Laird, K.Jones and T.Mitra, *Executing SOA: A practical Guide For The Service Oriented Architect*. Boston, MA: Pearson Education, 2008.
- [4] N.M.Josuttis, *SOA In Practice: The Art of Distributed System Design*, 1st ed. Sebastopol, CA: O'Reilly Media Inc, 2007.
- [5] D.Das, "PREMASA Model: An Integrated Approach To Customer Relationship Management," *IEM International Journal of Management & Technology*, vol.2, no.1, pp.111-118, Jan. 2012.
- [6] D.Krafzig, K.Banke, D. Slama, *Enterprise SOA: Service Oriented Architecture: Best Practices*. Upper Saddle River, NJ: Pearson Education, 2005.
- [7] D.Das, and P.Kundu, "Application of Service Oriented Architecture (SOA) in Implementing a CRM Process: A Conceptual Approach," - *Research and Higher Education in Computer Science and Information Technology*, S. S. Sau and A. D. Gupta, eds., Kolkata: SM, pp.148-152,2012.
- [8] A.D.Nghiem, *IT Web Services: A Roadmap for the Enterprise*. Upper Saddle River, NJ: Pearson Education, Inc., 2003.
- [9] W.Iverson, *Real World Web Services*. Sebastopol, CA: O'Reilly Media, Inc., 2004.
- [10] R.Nagappan, R. Skoczylas, and R.P. Sriganesh, *Developing JAVA Web Services: Architecting & Developing Secure Web Services Using JAVA*. Daryaganj, New Delhi: Wiley India (P) Ltd., 2004.
- [11] E.Cerami, *Web Services Essentials: Distributed Applications with XML-RPC, SOAP, UDDI & WSDL*. Sebastopol, CA: O'Reilly & Associates, Inc., 2012.
- [12] R.R. Branco, "Dynamic Program Analysis and Software Exploitation: From the crash to the exploit code," http://www.troopers.de/wpcontent/uploads/2011/04/TR11_Branco_Dynamic_program_analysis.pdf. 2011.
- [13] E. Cesar, A. Morajko, T. Margalef, J. Sorribes, A. Espinosa, E. Luque, "Dynamic performance tuning supported by program specification," - *Performance oriented application development for distributed architectures*, M.Gerndt, eds., Amsterdam: IOS Press, pp. 35-44, 2002.
- [14] R.E .Ladner, R.Fortna, B.H. Nguyen, "A Comparison of Cache Aware and Cache Oblivious Static Search Trees Using Program Instrumentation," - *Experimental algorithmics: from algorithm design to robust and efficient software*, R.Fleischer, B. M. E. Moret, & E.M.Schmidt, eds., Berlin: Springer -Verlag, pp.78-92, 2002.
- [15] M.Bach, M.Charney, R.Cohn, E.Demikhovsky, T.Devor, K.Hazelwood, A.Jaleel, C.K.Luk, G.Lyons, H.Patil, A.Tal, "Analyzing Parallel Programs with Pin," <http://www.cs.virginia.edu/kim/docs/ieeeComputer10.pdf>.2010.
- [16] P.Kundu, "Visualization of Program for Extracting Object Oriented Behaviour & Optimization: A Futuristic Approach," - *Research and Higher Education in Computer Science and Information Technology*, S. S. Sau and A. D. Gupta, eds., Kolkata: SM, pp.108-112,2012.
- [17] Prasenjit Kundu and Debabrata Das, "An Attempt to Analyze & Resolve the Pitfalls in CRM Software through Plug-In Instrumentation", *International Journal of Scientific and Research Publications*, vol. 2,issue 5, pp. 313-320, May. 2012.
- [18] D.Das, and P.Kundu, "Applications of Dynamic Program Analysis in a CRM Process: A Futuristic Concept," *International Journal of Scientific and Research Publications*, vol.2, issue. 4, pp. 306-318, Apr.2012.

-
- Prasenjit Kundu is currently pursuing doctoral degree program in computer science in Utkal University, India, PH-09831040996. E-mail: kundusantukundu@rediffmail.com
 - Debabrata Das is currently the faculty member in Bengal School of Technology & Management, India, PH-09874378766. E-mail: deb.brata.das@gmail.com
 - Bikram Kesari Ratha is currently the faculty member in the department of computer science in Utkal University, India, PH-09403361904. E-mail: vkramus@yahoo.com