International Journal of Computer and Electrical Engineering, Vol.3, No.1, February, 2011
1793-8163

# An Effective Service Oriented Model for Power System Transient Stability Analysis

M.Balasingh Moses, V. Ramachandran and P.Lakshmi

*Abstract*—**The power systems industry has unprecedented changes in its structure every day. In the open market environment, entities like generation, transmission and distribution make new issues in power system operations and planning are inevitable. One of the major consequences of this environment is the greater emphasis on stable operation of power systems. This paper aims at the development of Service Oriented Architecture (SOA) based model for representing large interconnected power systems and its operations. An attempt has been made to test the proposed model for transient 'stability analysis. The proposed SOA model includes Pre-Fault, During-Fault, Post-Fault and Swing Curve services which are accessible to the power system clients when the system is subjected to large disturbances. The stability services are configured and deployed in the service provider and then published in a registry for enabling universal visibility and access to the power system clients. A generalized XML based model for data representation has been proposed for exchanging data in order to handle interaction legacy power system applications in a service oriented environment. An effective SOAP based communication model is proposed to bind the power system stability services. The proposed model portraits the stability services in service oriented environment that provides a flexible, scalable and reliable infrastructure for representing and analysis of power system operations.**

*Index Terms*—**Stability, SOAP, SOA, WSDL and XML**

## I. Introduction

Stability is an important constraint in power system operations [1]. The computer applications used in power system operations have undergone profound changes since the last decade. The existing power system operations are primarily desktop applications with a small number of exceptions implemented in parallel processing super computers [2]. The simulation package for analysing the power system stability problems are programmed in FORTRAN or C. It has been known the software systems developed using platform dependent paradigms are expensive to maintain, difficult to extend and hard to integrate into other systems. Later, these systems have been replaced using object oriented and component based technologies. In these technologies, the power system data and operations are encapsulated as objects those can interact with each other by sending messages between power system applications. To monitor the power system operations, Supervisory Control and Data Acquisition (SCADA) and Energy Management System (EMS) are being developed [3]. The systems provided by different power sector vendors run on different hardware and software platforms. The conventional client-server architecture for power system analysis is complicated, memory management is difficult, source code is bulky, and exception-handling mechanism is not so easy [4-6]. Several models have been reported for analysing the large interconnected power systems require a common computational environment for representing its data and operations [7-11]. Since the existing power system is highly complex and widely distributed, it is needed to develop service oriented open system made up of a variety of power system services operating on dissimilar platforms, so that more extensive data and applications can be shared easily and flexibly. A distributed computing environment has to be built, so that the power system operations can be performed using geographically distributed resources [12-13]. A loosely coupled environment is required that allows other services to be plugged-in easily in order to solve the scalability issues in power systems.The software architecture had changed from central to distributed and distributed to Web-based, and now to Service Oriented Architecture. The proposed SOA model for carrying out power system stability operations will provide powerful set of features based on open standards like XML, SOAP, WSDL and UDDI. The model supports interoperability between services on different platforms in a reliable, scalable and adaptable manner inherently.

Manuscript received May 27, 2010.

M.Balasingh Moses is with the research scholar of Anna University, Chennai. phone: +91 94441 50585;( e-mail: balasinghmoses@gmail.com)

Dr.V.Ramachandran is with Professor of Department of Information Science and Technology, Anna University, Chennai. (e-mail: rama@annauniv.edu).

Dr.P.Lakshmi is with the Associate Professor of Electrical Engineering Department, Anna University, Chennai (e-mail:p_lakshmi@annauniv.edu)

## II. XML Representation for Power System Stability Data

The IEEE recommended a common data format for exchanging power system data which is not flexible and is limited to static data and not suitable for storing dynamic information. The modern power system needs more flexible and well-documented format for its data representation and exchange. Rapidly growing power sector environment changes their data structure or adds new data structure whenever new participants join in the power network. The data exchange must have a protocol which makes the data meaningful for each power system operation [17]. The XMLised representation of power system data offers reliable data exchange between legacy power system applications.

Certain data are common to all power system applications. All clients in the interconnected power systems have to know the general specification of the system. The base value of the system, number of buses, number of generators, transmission

lines, transformers, acceleration factor and tolerance value are classified as general data. The XML representation of general data required for power system analysis is as follows:

```
<GENERAL>
<Base MVA>Base value of the system</Base MVA>
<NB> Number of Buses </NB>
<NG>Number of Generators</NG>
    <NL>Number of Lines</NL>
<NT>Number of Transformers</NT>
<Alpha>Acceleration factor</Alpha>
<Tolerance> Tolerance value</Tolerance>
</GENERAL>
```

The power system line data includes transformer line rating which is essential for reliable planning and operation of the interconnected systems. This rating incorporates values for resistance, reactance, off line charging admittance and acceptable electrical loading on equipment before, during and after system disturbances. The way in which the lines are connected between the buses is also stored in the line data. The XML representation of line data required for stability analysis is as follows:

```
<LNEDATA>
    <SB> Sending Bus </SB>
    <RB> Receiving bus </RB>
    <R> Resistance of the line </R>
    <X> Reactance of the line </X>
    <B> Off line Charging admittance <B>
    <Rating> Maximum rating of the line </Rating>
<LINEDATA>
```

The transmission line data would have to be updated based on weather and seasonal conditions.The generation capacity and the consumer demand are major factors for analysing the stability of the power systems. The power system bus data includes magnitude and phase angle of voltages, real and reactive power flow on the interconnected systems and the rating of shunt capacitor. The operating limits of the buses have to be represented for the violations of the power system operations. The XML representation of bus data required for stability analysis is as follows:

```
<BUSDATA>
    <SLACK_ BUS>
     <VOLTAGE> voltage at the bus</VOLTAGE>
     <ANGLE>Load angle of the Bus </ANGLE>
    </SLACK_ BUS>
    <GENERATOR_ BUS>
<MW> Real Power </MW>
<VOLTAGE>Generating Voltage</VOLTAGE>
    </GENERATOR_ BUS>
    <LOAD_ BUS>
<MW> Real Power </MW>
<MVAR>Reactive Power </MVAR>
    </LOAD_ BUS>
    <LIMITS unit='MVAR'>
<MAX>Maximum Réactive Power Limit</MAX>
<MIN>Minimum Réactive Power Limit </MIN>
    </LIMITS>
</BUSDATA>
```

The synchronous machine data are essential to ensure stable operation of the power systems. The generating capability to meet the projected demands is the most important factor in the transient stability analysis. The synchronous machine data emphasize the values of synchronous reactance, synchronous speed, moment of inertia, number of poles and damping factor. The XML representation of synchronous machine data required for stability analysis is as follows:

```
<MACHINEDATA>
    <XS> Synchronous reactance </XS>
    <W> Synchronous speed </W>
    <N> Number of poles </N>
     <J> Moment of Inertia </J>
    <D>Damping factor </D>
</MACHINEDATA>
```

The conversion of power system data in to an XML form enables the independency of the data used various power system applications. The proposed XMLised power system data representation model significantly reduces the engineering efforts required to integrate its data in the service oriented environment.

## III. Proposed SOA Model for Transient Stability Analysis

The proposed Service Oriented Architectural model for representation of power system stability services is shown in Fig 1. The SOA model has three elements namely Stability Service Provider, Power Systems Registry and Client.

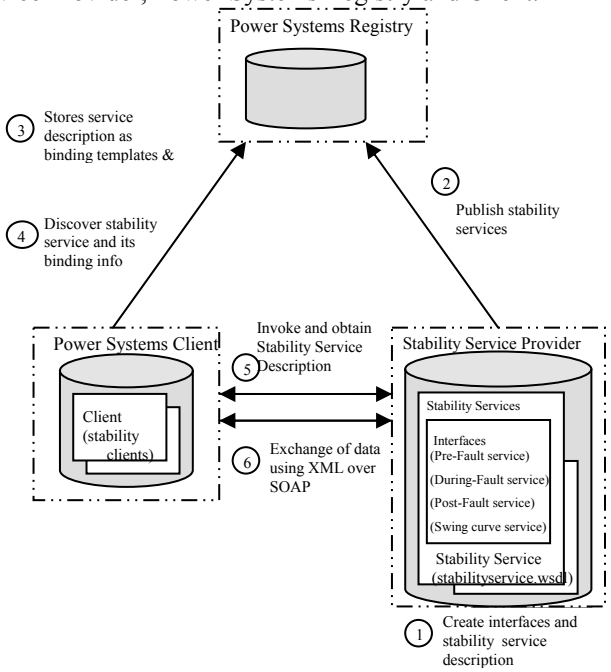


Fig 1. Proposed SOA model for Transient stability Analysis

### A. Stability Service Provider

The service provider is responsible for developing and deploying the stability services. The stability services are categorized in to Pre-Fault, During-Fault, Post-Fault and Swing curve services. A Stability Service provider offers the

above services and describes the interface information of the services using Web Service Description Language (WSDL) and as Stability Service Descriptors(SSDs) which is in the form of XML. The service provider makes the services available in the Power System Registry. The elements and relationship of the services are shown in Fig 2.

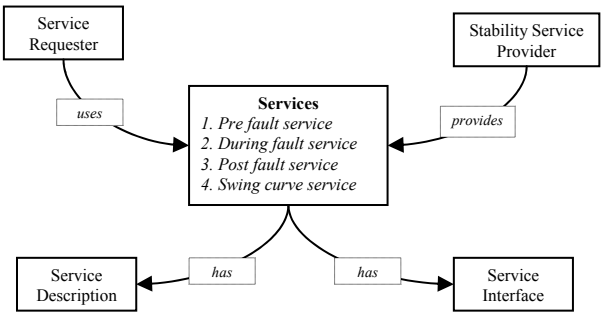


Fig 2. Stability Services

Stability services are self-describing, open components that support rapid, low-cost composition of distributed applications. The stability service provider supplies the service descriptions and implements the stability services. A service is a reusable function that can be invoked by another component through a well-defined interface. Services are loosely coupled, that is, they hide their implementation details and only expose their interfaces. In this manner, power system client need not be aware of any underlying technology or programming paradigm which the service is using. The loose coupling nature between services allows for a quicker response to changes than the existing conventional applications for power system operations.

*B. Power System Registry*

A power system registry is an authoritative, centrally controlled store of information about the stability services. Registry has to delegate permission to approved provider entities that wish to publish their own service descriptions. The main purpose of the registry is to allow fast and reliable communication and interoperability among diverse applications with minimal human oversight.The registry allows the power system client to efficiently discover and communicate with the services through the URLs.

*C. Power System Clients*

The power system clients discover the service available in the registry by service names and acquire the interface information using Stability Service Descriptors. The communication between the client and provider is shown in Fig 3.

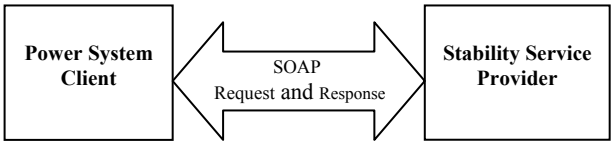


Fig. 3. Client Communication

Based on this information, the clients have a binding with the stability service provider and can invoke services using Simple Object Access Protocol (SOAP).

## IV. Implementation Of Proposed SOA Model

In the large interconnected systems, the client can invoke any of the published services which are required at the time of power system operations through well-defined interfaces. The various stages involved in the implementation of proposed SOA model for Stability analysis are service interfaces, service description, service configuration, service mapping, service publishing, service discovering and service invoking. The implementation details of these stages are explained in the following sections.

*A. Stability Service interfaces*

The service interface provides the contract between the power system client and stability service provider as shown in Fig 4. The service interface is responsible for all of the implementation details needed to perform the communication between the clients and service provider. In this proposed model, four interfaces are created for the representation of Pre-Fault, During-Fault, Post-Fault and Swing curve services.

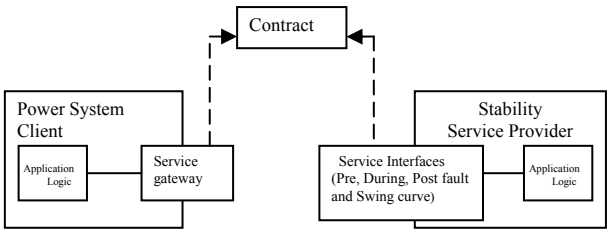


Fig. 4. Stability Service contract

Decoupling the service interface code from the service implementation code enables the system to deploy two codebases on separate tiers, potentially increasing the deployment flexibility. The service interface for computing swing curve is as follows:

```
package stability;
public interface swing  extends Remote
 {
 public String computeSwing() throws RemoteException ;
 }
```

The service interface encapsulates all aspects of the network protocol used for communication between clients and service provider. All other interfaces are described similarly.

*B. Stability Service Descriptors (SSDs)*

WSDL is used as the metadata language for defining the stability services. It describes how the service provider and client communicate with each other. WSDL is capable of describing services that are implemented using any language and deployed on any platform. It represents information about the interface and semantics of how to invoke a service. It contains the information about the data type, binding and address information for invoking the services from the

service provider. The Swing service is described as follows:

```
<definitions name=”stabilityservice” targetNamespace
                                        ="urn:Stability">
 <types>
   <schema targetNamespace="urn:Stability"
    xmlns:tns= "urn:Stability"xmlns: soap11- enc=
    "http://schemas.xmlsoap.org/soap/encoding/" >
   <complexType name="computeSwing">
   </complexType>
   <element name="computeSwing" type=
                              "tns:computeSwing"/>
   <element name="computeSwingResponse" type
                                        ="string"/>
 </types>
 <message name="swing_computSswing">
    <portType name="swing">
    <operation name="c eSwing">
   <input message="tns:swing_computSwing"/>
   <output
       message="tns:swing_computeSwingResponse"/>
  </operation>
  </portType>
  <binding name="swingBinding" type="tns:swing">
 <soap:binding transport= "http://
   schemas.xmlsoap.org/soap/http" style="document"/>
 <operation name="computeSwing">
  <soap:operation soapAction=""/>
  <soap:body use="literal"/
  </output>
  </operation></binding>
  <service name="Stabilityservice">
 <port name="swingport"
                        binding="tns:swingBinding">
    <soap:address
       location="REPLACE_WITH_ACTUAL_URL"/>
    <soap:address
       location="REPLACE_WITH_ACTUAL_URL"/>
   </port>
  </service>
</definitions>
```

The WSDL descriptor document consists of seven key structural elements for describing stability service. The <definitions> element defines the name of the service as ‘stabilityservice’ and declares the namespace as ‘Stability’. The <types> element defines the data types that would be used to describe the Bus and Line data. The <message> element represents a logical definition of the data being transmitted between the client and the service provider .The <portType> element provides the abstract definition of the operation (computeSwing) of the service, request and response messages. The <binding> element specifies a concrete protocol (SOAP) used for representing messages. The <service> element represents the service to be invoked over multiple bindings. The <port> element specifies an address (swingport) for binding to the service.

*C. Configuring the Stability Services*

The services related to computing of power system stability are configured as follows:

```
<service  name="stabilityservice"
          targetNamespace="urn:Stability"
          typeNamespace="urn:Stability"
          packageName="stability">
   <interface name="stability.prefault"/>
   <interface name="stability.duringfault"/>
   <interface name="stability.postfault"/>
   <interface name="stability.swing"/>
</service>
```

This configuration file contains the information and details about the deployed stability services (prefault, duringfault, postfault, swing curve) and metadata such as their service name (stabilityservice), namespace (Stability) and description.

*D. Mapping the Services*

The mapping information is specified by an XML document. This document describes how the properties of the Swing object have to be translated and mapped into XML. The mapping information for Swing object is as follows

```
<package-mapping>
  <package-type>stability</package-type>
  <namespaceURI>urn:Stability</namespaceURI>
  </package-mapping>
  <java-xml-type-mapping>
   <java-type>stability.swing_computeswing_
                    RequestStruct </java-type>
   <root-type-qname xmlns:typeNS="urn:Stability">
         typeNS:computeswing</root-type-qname>
   <qname-scope>complexType</qname-scope>
  </java-xml-type-mapping>
  <port-mapping><port-name>swingPort</port-name>
   <java-port-name>swingPort</java-port-name>
  </port-mapping>
  <service-endpoint-interface>stability.swing
  </service-endpoint-interface>
 <wsdl-port-type xmlns:portTypeNS="urn:Stability">
              portTypeNS:swing</wsdl-port-type>
 <wsdl-binding xmlns:bindingNS="urn:Stability">
         bindingNS:swingBinding</wsdl-binding>
  <service-endpoint-method-mapping>
   <java-method-name>computeSwing</java-method-
                                        name>
  </service-endpoint-method-mapping>
```

The mapping file describes the elements like package, type, port, method, and endpoint. The mapping information for the other services are described similar manner.

*E. Publishing the Services in Registry*

The proposed SOA model for Stability Analysis requires the common registry to deploy the services for easy integration, reuse and effective governance of services to meet the requirements. The registry allows the power system client to efficiently discover and communicate with the services. The main purpose of the registry is to allow fast and reliable communication and interoperability among diverse

applications. The stability services which are available in the registry are shown in Fig 5.

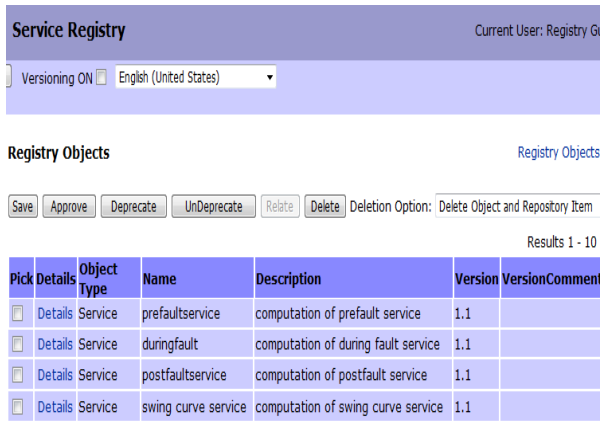


Fig. 5.stability services available in the Registry

*F. Discover the Stability Services*

At the time of operation, the power system clients discover the required service which is available in the Registry. Once the service is discovered, the client will query the services by their names to get the binding information and the identification of the provider. Based on this information, the clients access the provider for invoking the services. The procedure to discover the services is as follows

```
RegistryService rs = connection.getRegistryService();
BusinessQueryManager qm =
                    rs.getBusinessQueryManager();
BulkResponse response1 =
                    qm.getRegistryObjects("Service");
Collection c=object.getServiceBindings();
ServiceBinding sb=(ServiceBinding)oi.next();
String uri=sb.getAccessURI();
```

The procedure involves the establishment of connection to the registry, querying the stability service, getting the binding information and accessing URI to invoke a service.

*G. Binding the Stability Services*

The power system clients communicate with the stability service provider using SOAP message as shown in Fig 6. The XML form of Bus and Line data are attached as input parameter in the SOAP Body. The SOAP Body represents the mandatory processing information between the client and service provider. SOAP uses HTTP POST method for request and response in the form of messages.

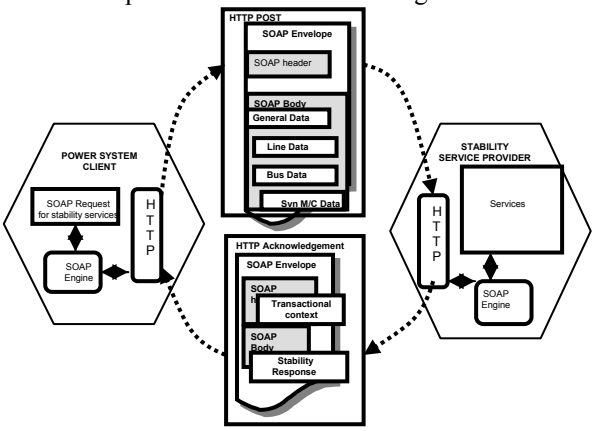


Fig. 6. SOAP communications between client and service provider

SOAP messages are configured for stability service invocation and response. The request message consists of IP address of the service provider and the required power system data in XML form. The following code segment delineates how general, bus, line and synchronous machine data have been attached to the SOAP message.

```
MessageFactory messageFactory=
                    MessageFactory.newInstance();
SOAPMessage message  =
                    messageFactory.createMessage();
SOAPPart soapPart = message.getSOAPPart();
SOAPEnvelope envelope = soapPart.getEnvelope();
SOAPBody body = envelope.getBody();
SOAPElement bodyElement =
          body.addChildElement(envelope.createName
          (operation,  "",  urn));
FileInputStream fs=new
                    FileInputStream("generaldata.xml");
FileInputStream fs=new FileInputStream("busdata.xml");
FileInputStream fs=new FileInputStream("linedata.xml");
FileInputStream fs=new
                    FileInputStream("machinedata.xml");
AttachmentPart attachment =
    message.createAttachmentPart(xmldata,"text/plain");
attachment.setContentId("generaldata");
message.addAttachmentPart(attachment);
```

A binding describes how the SOAP request and response messages have been sent through the transport protocol.

*H. Invoking Stability Services*

The power system clients are capable of invoking the required stability services as given in the SOAP message at the specified port is shown in Fig 6

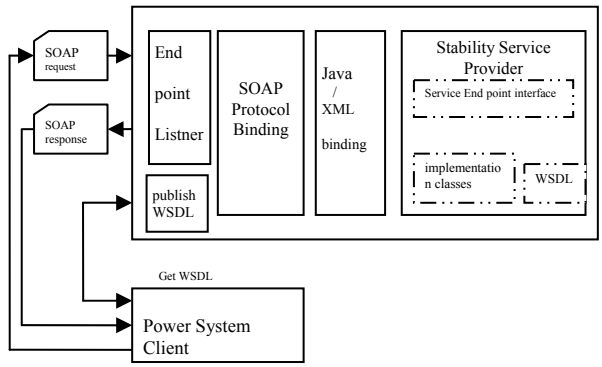


Fig. 6 . Invoking the Stability Services

The following code segment delineates how the swing service is being invoked.

```
String destination =
  "http:/192.168.1.1:1078/powersystem/swingservice";
SOAPConnectionFactory soapConnFactory =
                    SOAPConnectionFactory.newInstance();
SOAPConnection connection =
                    soapConnFactory.createConnection();
SOAPMessage reply = connection.call(message,
                                        destination);
```

SOAPPart soapPart = reply.getSOAPPart();
SOAPEnvelope envelope = soapPart.getEnvelope();
*SOAPBody body = envelope.getBody();*

The SOAP message contains the general, line, bus and synchronous machine data for invoking the stability services are shown below.

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV=
        http://schemas.xmlsoap.org/soap/envelope/>
<SOAP-ENV:Body>
 <compute xmlns="urn:TSA"/></SOAP-ENV:Body>
 </SOAP-ENV:Envelope>
 ------=_Part_0_9737354.1270895743906
  Content-Type: text/plain
  Content-ID: generaldata
   <generaldata>
   <tol>0.00001</tol>
   <alpha>1.4</alpha>
   <nb>6</nb>
   <ng>3</ng>
   <nl>11</nl>
   <clearingt>0.30</clearingt>
   </generaldata>
   <linedata>
     <line>
                  <from>1</from>
                 <to>4</to>
               --------
     </line>
   </linedata>
   <busdata>
    <pqbus>
                   <busno>2</busno>
                  <qmax>1.5</qmax>
                 <qmin>-0.9</qmin>
             <pload>-0.7</pload>
       <qload>-0.5</qload>
   </pqbus>
   </busdata>
   <machinedata>
   <machine>
                   <i>1</i>
                <xd>0.23</xd>
              <iner>10.0</iner>
              <freq>50.0</freq>
              <damp>12.0</damp>
   </machine>
   </machinedata>
```

## V. Results

The swing curve response obtained using the proposed SOA model is shown below

```
<env:Envelope
  xmlns:env="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:enc="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:ns0="urn:LoadFlow"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XML..-instanc e">
  <env:Body>
  <ns:computeResponse>
  <result>
  <swing curve>
   <deltalist>
       <delta>0.9554</delta>
      <delta>1.0054</delta>
                -----
   </deltalist>
  <time>
     <t1>0.1</t1>
     <t2>0.2</t2>
          -------
   </time>
  </result>
  </ns:computeResponse>
  </env:Body>
  </env:Envelope>
```

In the proposed SOA model, the various stability services can be invoked by the clients without any limitations.

## VI. Conclusion

An effective Service Oriented Architectural model has been developed for representing transient stability analysis of a large interconnected power system and tested for a sample of 14, 30 and 39 bus systems. This proposed model is scalable for any number of power system clients and the stability services can be invoked without any limitation in this service oriented environment. The various power system services can be plugged into this model and the services are made available anytime and anywhere for the power system operations.

## References

[1] P.kundur, J.Paserba, V.Ajjarpu, G. Anderson, A.Bose, C.Canizares, N.Hatziargyriou, D.Hill, A.tankovie, C. Taylor, T.V.Cutsem and V.Vittal, “Definition and classification of power system stability”, IEEE trans on power system, vol 19,no 3,Aug 2004, , pp.1387-1401

[2] Vuong G.T, Chahine R, Behling S.”Supercomputing for Power System Analysis", IEEE Transaction on computer application in power, Issue 3, August 2002, pp.45-49.

[3] B. Qiu and H.B.Gooi,“Web-Based SCADA Display Systems (WSDS) for Access via Internet”. IEEE Transaction on Power systems, vol 15, no 2, May 2000, pp. 681-686

[4] Shiming Tian Zhengme Xiang Tianmin Han, ”Large-Scale electric power system stability computation: a three level hierarchical model and algorithm ”,International conference on Advances in Power System Control, Operation and Management”, vol 1, December 1993, pp.114-120,.

[5] Cheng-Tsung Liu and Tzeng-Shong Yeh,” An inherent parallel algorithm for analyzing power system transient characteristics on a multiprocessor system”, Electric Power system Research, vol 1, Issue 2, March 1994, pp. 121-129

[6] Monika ten Bruggencate ,Suresh Chalasani,” Parallel mplementations of the power system transient stability problem on clusters of workstations“ International Conference on High Performance Networking and Computing, Article no 34, 1995

[7] Chen S, F.Y.Lu, “Web-Based Simulations of Power Systems” . IEEE Transaction on computer application in power, Issue 1, August 2002, pp.35-40.

[8] Vaahaedi E, Cheung K.W, Hydro B.C, Vancouver BC., ”Evolution and Future of On-line DSA” IEEE power engineering society, vol 1, July 1999, pp.291-293.

[9] Lavoie,V. Qué-Do ,J. L. Houle ,J. Davidson,” Real-time simulation of power system stability using parallel digital signal processors” Journal of Mathematics and Computers in Simulation, Volume 38 , Issue 4-6, August 1995, Pages: 283-292

[10] Hiyama T, Suzuki N, Kita T, Funakoshi T, " Development of Real Time Stability Monitoring system for Power system operation", IEEE power engineering society, vol.1, Feb 1999, pp. 525 – 530.
[11] Hong Chen Claudio A. Canizares Aajit Singh," Web based computing for power system Applications", North American Power Symposium, California, October 1999.
[12] V C Ramesh," On distributed computing for on-line power system Applications", International Journal of Electrical Power & Energy Systems, vol.18, Issue 8, November 1996, Pages 527-533.
[13] Aloisio, G. Bochicchio, M.A. La Scala, M. Sbrizzai, R.Lecce Univ," A distributed computing approach for real-time transient stability analysis",IEEE transactions on power systems, vol.12, Issue 2, May 1997, pp. 981-98
[14] Nagae, Yasutaro, Hirona, Takasaki, HigashidaI," Software Technologies toward the Network Era. Emergency Stability Control System for Large-Capacity Interconnected Power System "Science Links Japan, vol 53, no 8,1998, pp. 47-50
[15] Daniel Ruiz, R.Messina and Mania pavella "Online Assessment and control of Transient oscillations Damping" ", IEEE Transaction on Power Systems, volume 19, no. 2, May2004, pp.1038-1046
[16] Nihad M, Afaneen and Ahmed, "Computer Aided Transient Stability Analysis",Journal of Computer Science, Volume 3, no.3, 2007, pp.149-153
[17] Eric comer, "Understanding Web Services: XML, WSDL, SOAP, and DDI"Addison Wesley, 2002
[18] Robert Skoczylas and Ramesh Nagappan, "Developing Java Web Services", Willey 2003.
[19] Douglas K. Barry, "Web service and Service oriented Architecture", Morgan Kaufmann Publisher, 2003
[20] Anderson P.M and Fouad A.A "Power System Control and Stability", Wiley Inter science, 2003

**Balasingh Moses M** is currently working as a Assisitant Professor in the Department of Electrical and Electronics Engineering, Anna University Tiruchirappalli, INDIA. He had received his Bachelor of Engineering (Electrical and Electronics Engineering) from Bharathidasan University, Tiruchirappalli and Master of Engineering (Power System Engineering) from Anna University, Chennai, INDIA in 1997 and 2004 respectively. He is currently doing Ph.D in Anna University Chennai, INDIA. His research interests include Power System Studies, Distributed Computing and Web Services.

**Ramachandran V** is currently working as a Professor of Computer Science and Engineering in College of Engineering, Guindy, Anna University, Chenai, INDIA. He has received his Masters of Engineering and PhD in Electrical Engineering from College of Engineering, Guindy, Anna University, Chennai, INDIA. His research interests include Power Systems Reliability Engineering, Network Security, Component Technologies, Soft Computing and Web Services.

**Lakshmi P** is currently working as a Assistant Professor of Electrical and Electronics Engineering in College of Engineering, Guindy, Anna University, Chenai, INDIA. She has received her Masters of Engineering and PhD in Electrical Engineering from College of Engineering, Guindy, Anna University, Chennai, INDIA. Her research interests include Power Systems Stability, Power Quality and Intelligent Controllers.