Engineering Letters, 13:3, EL_13_3_24 (Advance online publication: 4 November 2006)

Web Service Model for On-Line Load Flow Monitoring of Multi-Area Power Systems

R. Ramesh and V.Ramachandran

***Abstract*—Web services, and more in general service-oriented architectures (SOA), are emerging technologies of choice for implementing distributed architectural models for performing power system operations such as on-line load flow monitoring of multi-area power systems in a complete secure, distributed and platform independent environment. On-line load flow monitoring requires the calculation of power flow solutions by using real-time data obtained from the power system clients. The power system client applications are running in a heterogeneous environment. The proposed SOA model for on-line load flow monitoring is highly distributed and has inherent features such as scalability, reliability and also uses available computing power, hence economic feasibility is taken care implicitly.**

***Index Terms*— Distributed Systems, On-line Load flow monitoring, Service-Oriented Architecture Web services.**

I. Introduction

The power system operation and control needs huge volume of data and they have been distributed in a heterogeneous environment. Hence a new approach is required to enable the power system data to be processed, analyzed and interpreted by different power system clients. Existing power system simulations are primarily desktop applications with a small number of exceptions implemented on parallel processing computers. The existing Web enabled models for power system operations are mainly concerned with exchange of information and do not provide convenient environment to solve power system problems [1]. The rapid developments of the Internet, Web service and distributed computing have opened the door for feasible and cost effective solutions for multi-area power system problems. The existing web enabled models for solving multi-area power system problems are highly distributed and are operating in a heterogeneous environment, need to have an access to massive dataset and to do complex computations.

Manuscript received April 10, 2006

R.Ramesh is with the department of Electrical and Electronics Engineering, College of Engineering, Gunidy, Anna University, Chennai, India. (e-mail: rramesh@ annauniv.edu).

V.Ramachandrann was with Professor of Computer Science and Director of Ramanujan Computing Centre, Anna University, Chennai India. (e-mail: rama@ annauniv.edu).

The Web protocols are completely vendor-, platform-, and language-independent. Web services support Web-based access, easy integration and service reusability. With service-oriented architecture, everything is a service, encapsulating behavior and providing the behavior through an interface that can be invoked for use by other services on the network. Services are self-contained, modular applications that can be described, published, located and invoked over the Internet.

Chen and Lu demonstrated the potential advantages of the Web as the platform for developing and deploying complex power system simulations by using the distributed technologies and model-view controller concepts [2]. Since the existing Supervisory Control And Data Acquisition Systems (SCADA) were developed based on different platforms using different languages, the authors Qui et al proposed an Internet based SCADA display system and tried to solve the legacy issues using Java Native Interface (JNI), which is really a tedious process [3]. An RMI (Remote Method Invocation) based single-server/multiple-clients architecture has been proposed by Nithiyananthan et.al in such a way that for every specific period of time, the remote server obtains the power system data simultaneously from the neighboring power systems which are the clients registered with the remote load flow server and the load flow solutions from the server have been sent back to the respective clients [4].

A complete Web based and platform independent power system simulation package with various analysis distributed in a clustered environment has been modeled by Irving et.al [5]. Sando et al. demonstrated through experimental results that on-line security analysis could be executed in lesser period of time even for large power systems [6]. In future every electrical generator will be equipped with computational and communication facilities. Grid computing can provide a relatively inexpensive new technology allowing the output of embedded generators to be monitored. The ability of grid enabled systems to interact autonomously will be vital for small generators where manned operation is unlikely to be viable [7].

Jun Zhu stated that the Web services based integration of power system is much easier than other integration approaches and more legacy applications will participate in the integrated system [8]. The authors Quirino presented the definition and a first implementation of a Web service platform for performing efficient and scalable on-line power system security analysis [9]. Chun-Lien Su et.al explained the benefits of Internet interface to the SCADA systems that provides a favorable solution for data exchange [10]. It brings substantial benefits, including cost saving, better support, the availability of infrastructure components for building networks and access to widely accepted standard network protocols. Even though, there are many architectural models for solving multi-area power systems, the proposed model is more powerful and flexible since it provides standard technology and platform for heterogeneous environments.

II. NEED FOR WEB SERVICE MODEL FOR ON-LINE POWER SYSTEM ANALYSIS

The integration of legacy power system applications is a challenging task. These legacy applications were developed when there were no open standards. Vendors developed and deployed these legacy systems using their proprietary technologies that are not interoperable with each other. To address these integration issues many on-line power system models were developed that would facilitate the integration of heterogeneous power system applications. The Utility Integration Bus (UIB) provides a common information-bus-based integration model for data exchange and communication between utility information systems. It simplifies the integration of loosely coupled applications by replacing the traditional point-to-point proprietary interfaces with a universal API, which provides access to all participating applications [11]. While these specifications and initiatives successfully address some of the strategic integration issues, implementation of these good integration strategies remains a challenging task. The traditional Enterprise Application Integration (EAI) approaches normally involve developing middleware applications to communicate with the non-interoperable applications using message broker technology. Developing and deploying a compatible object request broker (ORB) with CORBA or DCOM is a fairly complex issue, a task requiring special expertise. The achievement of a good integration strategy is significantly constrained by many implementation restrictions in the traditional EAI solutions, such as application interoperability and implementation complexity, etc. Web services have emerged as the next generation of integration technology. Based on open standards, the Web services technology allows any piece of software to communicate with each other in a standardized XML (eXtensible Markup Language) messaging system. It eliminates many of the interoperability issues that the traditional EAI solutions have difficulty resolving. The underlying transport protocol behind Web services is based on XML over HTTP (Hyper Text Transfer Protocol). With minimal programming, the Web services technology enables the proposed model to easily and rapidly wrap the legacy power system applications and expose their functionality through a uniform and widely accessible interface over the Internet. The Web services are built on service-oriented architecture that operates effectively over the Web using XML-based protocols. The conventional client-server architecture model for power system analysis is complicated, memory management is difficult, source code is bulky and exception-handling mechanism is not so easy. In the conventional; power system operation and control, it is assumed that the information required for the monitoring and controlling of power systems is centrally available and all computations are to be done sequentially at a single location. With respect to sequential computation, the server has to be loaded every time for each client's request and time taken to deliver the solution is also relatively high. So the service-oriented architectural model is essential for power system applications in order to solve legacy issues and maintain interoperability. The service-oriented architecture model for on-line load flow monitoring of multi-area power systems is given in Fig.1

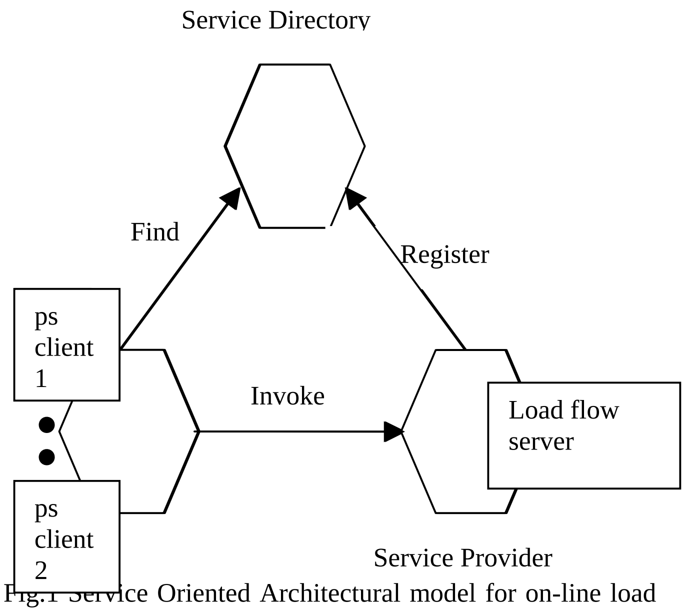


Fig.1 Service Oriented Architectural model for on-line load flow monitoring of multi-area power systems

The load flow service has been designed and implemented in the server side container. A service provider makes the load flow service available and publishes the contract that describes its interface. It then registers the load flow service with a service broker. Any power system client as service consumer queries the service broker and finds load flow service. The service broker then gives the directions to the power system client regarding where to find the load flow service and its service contract. The power system data has been represented in XML and XMLised data is packaged in a SOAP envelope and sent over HTTP transport. The proposed model is interoperable since the power system data are represented in XML. The Power System Web Service Description (PWSDL) module has been developed to create service facilitators and to implement proxies to establish connection between power system client and load flow server.

III. WEB SERVICE MODEL FOR ON-LINE LOAD FLOW MONITORING OF MULTI-AREA POWER SYSTEMS

In the proposed model, the procedure for load flow computation is implemented as a Web service in Java platform, while the power system clients which are requesting load flow monitoring have been implemented in different platforms. The load flow service is implemented using JAX-RPC (Java API for XML based remote procedure calls) as a Web service in a distributed environment. Conversely the power system client which is deployed in Java platform can communicate with the load flow service that was developed and deployed using some other paradigm. The built-in API for JAX-RPC allows the transmission of power system data in XML format. The XMLised power system data is enclosed in a SOAP envelope and can be communicated to the load flow Web service through PWSDL interface. The power system client needs only the PWSDL to access the load flow service.

The power system client can access the deployed load flow service using a Service Endpoint Interface (SEI). The PWSDL has a reference to the load flow service End-point interface. The service endpoint interface (SEI) is a remote interface that declares the remote methods through which the power system client interacts with the load flow service using JAX-RPC environment. A proxy for the remote load flow service object is created on the client side and the clients can invoke the load flow service method directly using this proxy object. The data needed for this service has already been encapsulated in SOAP envelope and can be parsed and communicated to the load flow service. The power system client communication with load flow service is shown in Fig.2

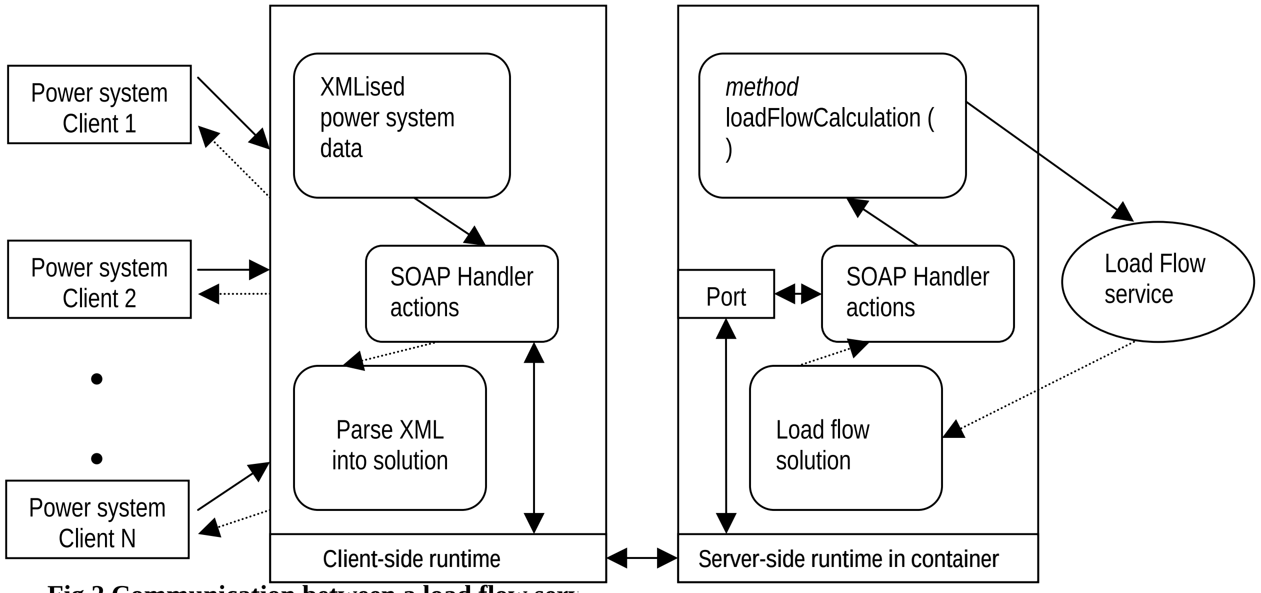


Fig.2 Communication between a load flow serv and power system client

The server side encompasses a runtime and a load flow service endpoint. The client side runtime contains components to convert power system data into XML form and to generate SOAP messages and a parser to convert the XML response into load flow solution. The remote procedure calls use an XML based protocol, such as SOAP as the application protocol and uses HTTP as the transport protocol. The JAX-RPC client and service runtime are responsible for sending and processing the remote method call and response respectively. The JAX-RPC client creates a SOAP message to invoke the remote method and the JAX-RPC service run time transforms the SOAP message to a Java method call and dispatches the method call to the service endpoint. The load flow Web service design and its implementation are discussed below.

A. *Load flow service Endpoint Interface*

The load flow service end point interface is defined as follows:

```
public interface loadFlowInt extends Remote
 {
    public String loadFlowCalculation ( String
            linedata, String busdata ) throws
                        RemoteException;
 }
```

B. *Load flow service Implementation*

The load flow service is implemented in a remote object which is an instance of the LoadFlowImpl class that implements the service end point interface (LoadFlowInt). The LoadFlowImpl class defines the method loadFlowCalculation, which is the required service. The power system client interacts with an object of class LoadFlowImpl by invoking the method loadFlowCalculation( ) through LoadFlowInt interface.

The method loadFlowCalculation( ) enables the power system client to get the load flow solution as a result. Fig 3 shows the sample code for load flow service implementation.

```
public class loadFlowImpl  implements loadFlowInt
{
  public String loadFlowCalculation (String
linedata,String
                         busdata)  throws RemoteException
       {
            // load flow calculation
       }
 }
```

Fig. 3 The sample code for load flow service implementation

C. JAX-RPC-mapping

One of the most important achievements of the JAX-RPC specification is its definition of a standard for mapping a WSDL document (which represents a service description) to its Java representation (service endpoints, stubs, ties, and Java types) and vice versa. The PWSDL has been created which provides an XML description of the load flow service that clients can invoke. The mapping file contains information that correlates the mapping between the Java interfaces and the PWSDL definition. The JAX-RPC mapping file match closely with the structure of a PWSDL file and provides mappings for PWSDL bindings, PWSDL port types and PWSDL messages. Fig.4 shows PWSDL code for on-line load flow service.

```
<?xml version =”1.0” encoding=”UTF-8”?>
  <message
name=”LoadFlowInt_loadFlowCalculation”>
        <part name =”linedata” type=”xsd:string”/>
        <part name =”busdata” type=”xsd:string”/>
    </message>
  <message  name=”LoadFlowInt_
           loadFlowCalculationResponse”>
              <part name =”solution” type=”xsd:string”/>
   </message>
  <binding name=”LoadFlowIntBinding”
                    type=”tns:LoadFlowInt”>
       <soap:binding                          transport=
”htto://schemas.xmlsoap.org/soap/http”     style=”rpc”/>
        <operation name=” loadFlowCalculation”>
        <soap:operation soapAction=””/>
<service name=”LoadflowService”>
     <port                      name=”LoadFlowIntPort”
binding=”tns:LoadFlowIntBinding”>

<soap:addresslocation=http://10.1.1.1:5864/loadflow-jax
rpc/loadflow
xmlns:wsdl=http://schemas.xmlsoap.org/wsdl//>
</xml>
```

Fig. 4   PWSDL code for on-line load flow service

D. Deploying the Load flow service

The load flow service has been packaged and deployed. During the process of deployment, a Web archive file (WAR) has been created in which the required service file has been added. The load flow service endpoint interface, LoadFlowService.wsdl file and the mapping file have also been added. The load flow service has been deployed in J2EE server. After deploying, The PWSDL file of the deployed load flow service can be viewed by requesting the URL **http://10.1.1.1:8080/loadflow-jaxrpc/loadflow?PWSDL** and given in Fig 4

IV. Power System Web Client

The power system client application invokes the method *loadFlowCalculation*( )on the load flow server. The power system client makes the call through a stub that acts as a client proxy to the remote load flow service. The client stubs has been created from the configuration file as shown below

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration
   xmlns="http://java.sun.com/xml/ns/jax-rpc/ri/config">
  <wsdl
location="http://localhost:8080/loadflow-jaxrpc/loadflow?
WSDL"
      packageName="sstub"/>
</configuration>
```

The stub object has been created using the LoadflowService_Impl object as well as other needed runtime files such as serializers and value types have been created. The endpoint address that the stub uses to access the load flow service is set and then the stub casted into LoadFlowInt service endpoint interface. Finally, the loadFlowCalculation ( ) method is invoked. Fig. 5 gives the code for the client application.

```
private String endpointAddress;
public static void main(String argv[])  {
     try
        {
           Stub stub =
createProxy();stub._setProperty(javax.xml.rpc.Stub.ENDP
OINT_ADDRESS_PROPERTY,
"http://localhost:8080/loadflow-jaxrpc/loadflow");
           lf = (LoadFlowInt) stub;
           lf.loadflowcalculation(linedata , busdata);
        } catch (Exception ex) {  ex.printStackTrace();}
      private static Stub createProxy()
       {
     return (Stub) (new
LoadflowService_Impl().getLoadFlowIntPort());
      }
  }
```

Fig. 5 The code for the client application

V. The hierarchical steps to carry out the on-line load flow MONITORING:

- Start the J2EE server.

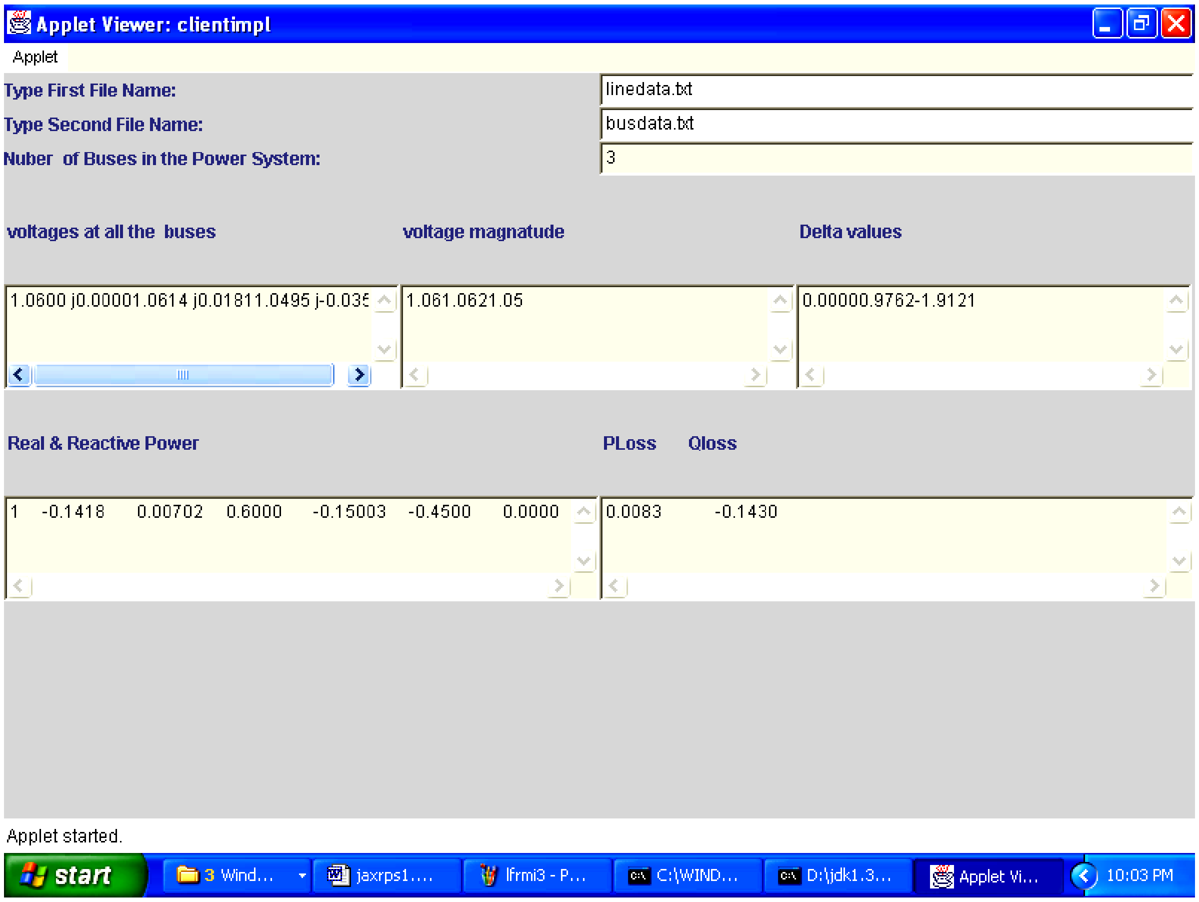
- Start the power system client. (Any number of clients may be considered.)
- Power system client can access the load flow service by requesting the URL http://10.1.1.1:8080/loadflow-jaxrpc/loadflow.
- Load flow server uses the power system client's reference to receive the load flow data from that client and computes the load flow solution.
- Client obtains the result and provides a view of the results through a user Interface.
- For every specific interval of time, the load flow server automatically receives load flow data from the client, thereby provides an automatic on-line load flow monitoring.

VI. TEST CASES AND RESULTS

The load flow analysis was carried out using the proposed model on a 3-bus system, 5-bus system, 6 -bus system and 9-bus system. The sample 3-bus system data in XML format are given in Table-1 and the load flow solution for 3-bus system have been shown in Fig 6.

```
<? xml version ="1.0" encoding="UTF-8"?>
 <bus>
    <busno>
      1
        <bustype>slack</bustype>
       <voltagespecified>1.06</ voltagespecified >
       <pgeneration>0.0</ pgeneration >
       <qgeneration>0.0</ qgeneration >
        <pload>0.0</pload>
        <qload>0.0</qload>
              </busno>
     <busno>
      2
        <bustype>pv</bustype>
       <voltagespecified>1.002</ voltagespecified >
       <pgeneration>-0.1</ pgeneration >
       <qgeneration>0.0</ qgeneration >
        <pload>0.0</pload>
        <qload>0.1</qload>
     </busno>
         .
        .
    </xml>
```

Table.1 Sample 3-bus bus power system data in XML format


Applet Viewer: clientimpl

Applet

Type First File Name: linedata.txt

Type Second File Name: busdata.txt

Nuber of Buses in the Power System: 3

voltages at all the buses

1.0600 j0.00001.0614 j0.01811.0495 j-0.035

voltage magnatude

1.061.0621.05

Delta values

0.00000.9762-1.9121

Real & Reactive Power

```
1   -0.1418     0.00702    0.6000     -0.15003    -0.4500     0.0000
```

PLoss   Qloss

```
0.0083        -0.1430
```

Applet started.

Fig 6. Load flow solution - Power system client's view

VII. CONCLUSION

The web service model is created using java and deployed in J2EE deployment tool. The efficient service oriented architectural model has been developed to carry out on-line load flow monitoring of multi area power systems. The system provides excellent scalability, high security and has a capacity to meet the huge computation requirement, which is suitable to carry out on-line load flow monitoring for large interconnected multi-area power systems. The practical implementation of this approach suggested in this paper has been assessed based on different sample power systems. The proposed model can be suitably implemented for a large power system network spread over a large geographical area

VII. REFERENCES

1. Qui.B. and Gooi.H.B. May 2000. "Web based SCADA display system for access via Internet" *IEEE Trans. On Power Systems*, 15, 681-686.
2. Chen.S. and Liu.F.Y. Jan 2002 " Web Based Simulations of Power Systems" *IEEE Trans. on Computer Applications in Power*, 15, issue 1 ,35-40
3. Bin Qiu et.al. Jan 2002. " Internet based SCADA display system" *IEEE Trans. On Computer applications in power*, 15, issue.1, pp14-19
4. Nithiyanatan.K and Ramachandran.V , winter-spring 2004."RMI Based Multi-Area Power System Load Flow Monitoring" *IJECE*, 3, no.1, pp.28-30,
5. Malcolm Irving, Gareth Taylor and Peter Hobson, 2004," Plug in to Grid Computing " *IEEE Trans. on Power and Energy Magazine*, 2, issue 2, pp 40-44.
6. M. Di Santo, N. Ranaldo, D.Villacci and E. Zimeo , 2004, " Performing Security Analysis of Large Scale Power Systems with a Broker-based Computational Grid " *IEEE Proceedings of ITCC2004*, 2,77-82
7. Malcolm Irving and Gareth Taylor , 2003 "Prospects for Grid-Computing in Future Power Network's" Discussion Paper circulated by Brunel Institute of Power Systems 15/09/03
8. Jun Zhu "A Web-Services-Based Framework for Integration of Power System Applications" *IEEE power & energy magazine* november/december 2003 pp. 40-49.
9. Quirino Morante1, Nadia Ranaldo, Eugenio Zimeo "Web Services Workflow for Power System Security Assessment"
10. Chun-Lien Su, Chan-Nan Lu and Tsun-Yu Hsiao "Simulation Study of Internet Based Inter Control Center Data Exchange for Complete Network Modeling" IEEE TRANSACTIONS ON POWER SYSTEMS, VOL. 17, NO. 4, NOVEMBER 2002 pp.1177- 1183
11. Mike P. Papazoglou "Service -Oriented Computing: Concepts, Characteristics and Directions" Proceedings of the Fourth International Conference on Web Information Systems Engineering (WISE'03) 2003

Ramesh received his M.E. degree in Power system engineering form the Faculty of Engineering and Technology, Annamalai University, Chidambaram, India in 2001. He is currently working as a lecturer in department of Electrical and Electronics Engineering, College of Engineering, Gunidy, Anna University, Chennai, India. His research interests include power system analysis and modeling, grid computing and Internet technologies.

Ramachandran received his M.E. degree and Ph.D. in Electrical Engineering form College of Engineering, Gunidy, Anna University, Chennai, India. He is currently working as a Professor of Computer Science and Engineering and Director of Ramanujan Computing Centre, Anna University, Chennai. His research interests include power system reliability engineering, network security , soft computing and Web technology.