Academic Open Internet Journal www.acadjournal.com Volume 19, 2006
ISSN 1311-4360

SERVICE-ORIENTED ARCHITECTURAL MODEL FOR ON-LINE LOAD FLOW MONITORING OF MULTI-AREA POWER SYSTEMS

Ramadoss **Ramesh** and Velimuthu **Ramachandran**
Ramanujan Computing Centre, Anna University, Chennai-25, India
E-mail: rramesh@annauniv.edu, rama@annauniv.edu

Abstract – Web services, and more in general service-oriented architectures (SOA), are emerging as the technologies and architectures of choice for implementing distributed architectural models for performing 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 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.

Keywords – On-line Load flow monitoring, Web service, service-oriented architecture

INTRODUCTION

The power system operation and control needs huge volume of data where a new approach is needed 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 [1] for power system operations are mainly concerned with exchange of information and do not provide quick and reliable solutions to power system problems. 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 [2] 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. Since the existing Supervisory Control And Data Acquisition Systems (SCADA) were developed based on different platforms using different languages, the authors Qui et al [3] 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. An RMI (Remote Method Invocation) based single-server/multiple-clients architecture has been proposed by Nithiyananthan et.al [4] 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.

A complete Web based, 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. [6] demonstrated through experimental results that on-line security analysis could be executed in lesser period of time even for large power systems. 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 [8] 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. The authors Quirino [9] presented the definition and a first implementation of a Web services based platform for performing efficient and scalable on-line power system security analysis. Chun-Lien Su et.al [10] explained the benefits of Internet interface to the SCADA systems that provides a favorable solution for data exchange. 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.

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. All these involve an access to heterogeneous, distributed, massive dataset and computation. Even though, there are many architectural models for solving multi-area power systems, the proposed service oriented architectural model provides a distributed solution for on-line load flow monitoring of multi-area power systems is more powerful and flexible since models provides standard technology and platform for heterogeneous systems.

NEED FOR WEB SERVICE MODEL FOR ON-LINE POWER SYSTEM MODEL

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. 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 enterprise 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. So the service-oriented architectural model is essential for power system applications in order to solve legacy issues and maintain interoperability. The service-oriented architectural model for on-line load flow monitoring of multi-area power systems is given in Figure .1

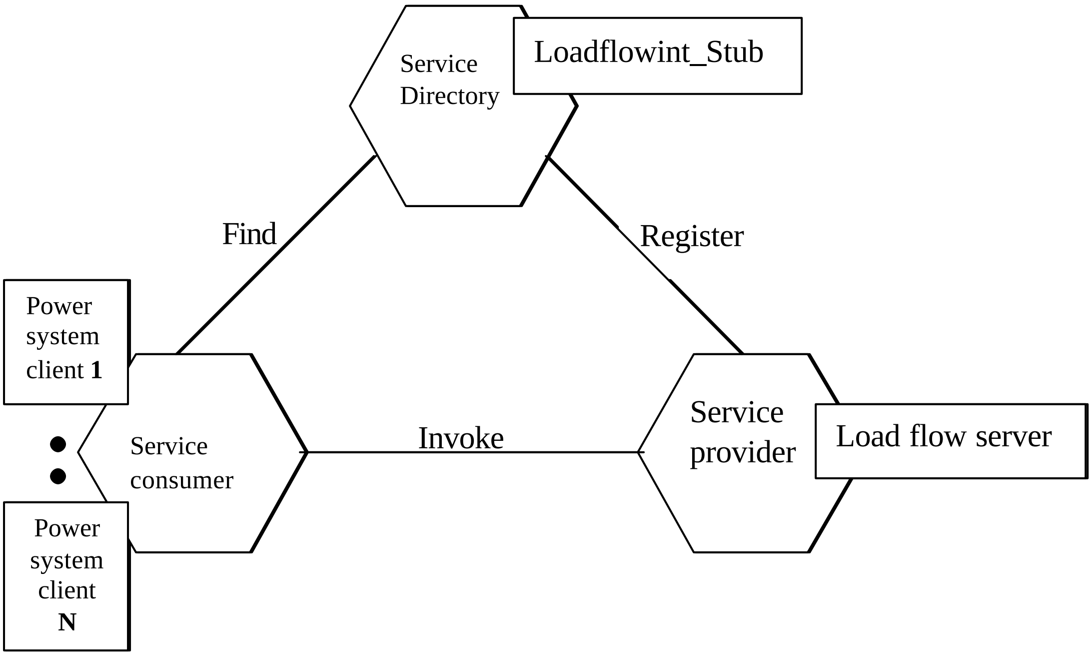

Figure.1 Service Oriented Architectural model for on-line load flow monitoring of multi-area power systems

The load flow monitoring has been designed as a load flow service and service provider makes the load flow service available & publishes. It then registers the load flow service with a service broker. Any power system client as service consumer queries the service broker then the service broker gives the directions to the power system client regarding where to find the load flow service.

Features of proposed model:

- Java RMI use optimized connection-oriented communications protocols that are either language specific, or have detailed rules defining how data-structure and interfaces should be realized. In contrast, Web services are based on the ubiquitous technologies that have grown up to support WWW-services (human-via-browser-to-application).

- In RMI mechanism, the client can communicate with server only if the stub is present and the availability of stub should be taken care by the developer. But in the case of web service, automatically it takes care the availability of stub.
- There is no concept of an object reference, instead a service is defind by an end-point that supports various operations.
- Web services provide interoperability between various software applications running on disparate platforms
- Web services use open standards and protocols. Protocols and data formats are text-based where possible, making it easy for developers to comprehend.
- By utilizing HTTP, web services can work through many common firewall security measures without requiring changes to the firewall filtering rules. Other forms of RPC may more often be blocked
- Web services allow software and services from different companies and locations to be combined easily to provide an integrated service
- Web services allow the reuse of services and components within an infrastructure.
- Web services are loosely coupled thereby facilitating a distributed approach to application integration

PROPOSED SERVICE-ORIENTED ARCHITECTURAL MODEL FOR ON-LINE LOAD FLOW MONITORING OF MULTI-AREA POWER SYSTEMS

In the proposed model, the power system data has been represented in XML which is a simple tag-based approach provides a flexible extensible mechanism that can handle the range of digital data from highly structured database records to unstructured. This feature is quite suitable for power system data exchange, since power applications are normally based on various platforms protocols from different vendors and also needs big amount of data to be transferred for monitoring the multi-area power systems in real time. XML standard provides excellent opportunity for the information exchange among different systems without changing much of the existing system. XML technology has been used for easy exchange of load flow data between disparate power systems and applications. The XMLised data is packaged in a Simple

Object Access Protocol (SOAP) envelope and sent over HTTP transport. 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. The power system client can access deployed load flow service using a service endpoint interface. The client creates the local object that represents the remote load flow service through PWSDL then simply invokes the methods on the proxy. The power system client communication with load flow service is shown in figure.2

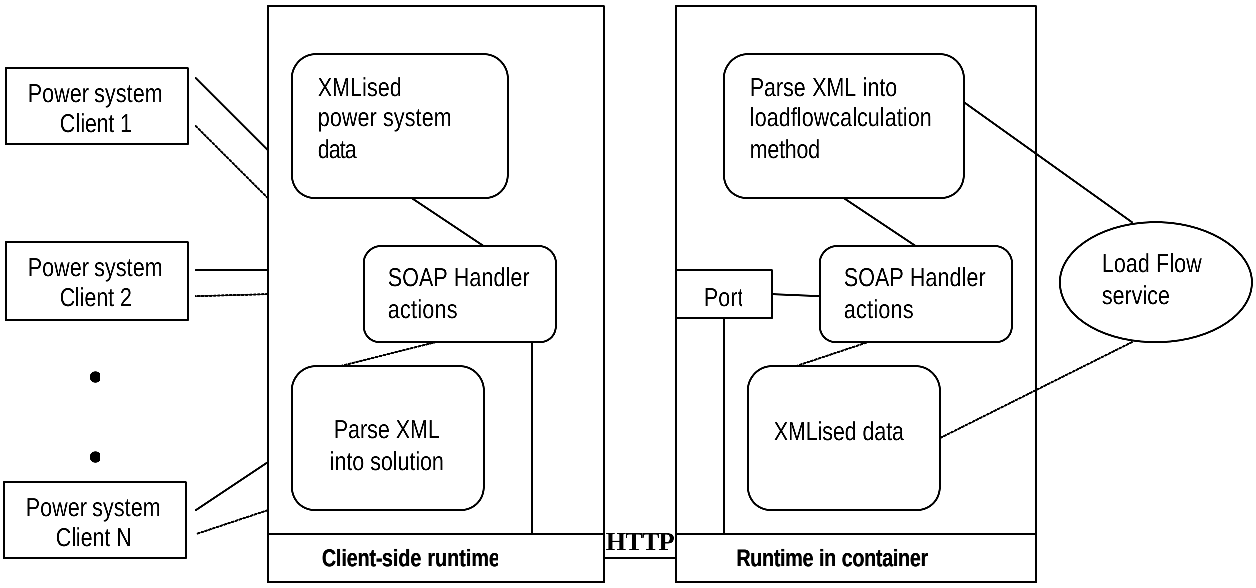

Figure.2 Communication between a load flow service and power system client

The server side contains a service runtime environment and a load flow service endpoint. The client side contains a client runtime environment and power system clients. The remote procedure calls use an XML based protocol, such as SOAP, as the application protocol, and they use HTTP as the transport protocol. The client creates a SOAP message to invoke the remote method and the service runtime transforms the SOAP message to a Java method call and dispatches the method call to the service endpoint.

Load flow service Endpoint Interface

The service endpoint interface (SEI) is defines the remote interface that describes the remote method available in load flow service through which the power system client interacts with the load flow service. The LoadFlowInt interface declares one method i.e. loadflowcalculation, which takes two values as a string and returns a value as string representing the load flow solution of the respective power system client.

Load flow service Implementation

After defining the remote interface, the load flow service implementation has been defined. Class **LoadFlowImpl** is the Web service endpoint that implements the **LoadFlowInt** interface. The service client interacts with an object of class **LoadFlowImpl** by invoking method **Loadflowcalculation** of interface **LoadFlowInt.** Method **Loadflowcalculation** enables the power system client to get the load flow solution as a result. Figure 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
       }
}
```

Figure 3 shows the sample code for load flow service implementation.

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. This mapping file provides the mappings for PWSDL bindings, PWSDL port types and PWSDL messages. Figure5 shows PWSDL code for on-line load flow service

```
    <?xml version =”1.0” encoding=”UTF-8”?>
  <message  name=”LoadFlowInt_call”>
      <part name =”linedata” type=”xsd:string”/>
      <part name =”busdata” type=”xsd:string”/>
   </message>
  <message  name=”LoadFlowInt_callResponse”>
      <part name =”result” type=”xsd:string”/>
   </message>
<port Type name=”LoadFlowInt”>
     <operation name=”call” parameterOrder=”String_1 String_2”>
           <input message”tns:loadflowint_call”/>
            <output message=”tns:LoadFlowInt_callResponse”/>
    </operation>
</portType>
<binding name=”LoadFlowIntBinding” type=”tns:LoadflowInt”>
        <soap:binding        transport=”htto://schemas.xmlsoap.org/soap/http”
style=”rpc”/>
        <operation name=”call”>
        <soap:operation soapAction=””/>
<service name=”MyLoadflowService”>
     <port name=”loadflowintPort” binding=”tns:loadflowintBinding”>
         <soap:addresslocation=http://10.1.1.1:5864/loadflow-
jaxrpc/loadflow   xmlns:wsdl=http://schemas.xmlsoap.org/wsdl//>
</xml>
```

Figure5 shows PWSDL code for on-line load flow service

Deploying the Load flow service

The load flow service has been deployed. 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?WSDL** as in Figure 4 .

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 in figure.7

```
<?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>
```

Figure.7 Configuration file

The stub object has been created using the MyLoadflowService_Impl object as well as other needed runtime files such as serializers and value types has been created. The endpoint address that the stub uses to access the load flow service is set and then the stub is cast to the LoadFlowInt service endpoint interface. Finally, the loadflowcalculation method is invoked. Figure.8 gives the code for the client application.

```
private String endpointAddress;
  public static void main(String argv[]) {
    try {
      // Invoke createProxy() to create a stub object
      Stub stub = createProxy();
      // Set the endpoint address the stub uses to access the service
      stub._setProperty(javax.xml.rpc.Stub.ENDPOINT_ADDRESS_PROPERTY,
                    "http://localhost:8080/loadflow-jaxrpc/loadflow");
      // Cast the stub to the service endpoint interface (LoadFlowInt)
      lf = (LoadFlowInt) stub;
        // Invoke the add method
       lf.loadflowcalculation(linedata , busdata);
               } catch (Exception ex) {
      ex.printStackTrace();
    }
  }
  private static Stub createProxy() {
    // Create a stub object
    // Note that MyFirstService_Impl, generated by wscompile, is implementation-
specific
    return (Stub) (new MyLoadflowService_Impl().getLoadFlowIntPort());
  }
 }
```

HIERARCHAL STEPS INVOLVED IN EXECUTION OF LOAD FLOW GRID SERVICE

- 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.

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.

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
12. java.sun.com