Internet, Open-source and Power System Simulation

Michael Zhou and Shizhao Zhou

(www.interpss.org)

***Abstract*--This paper presents an overview of the InterPSS project - an Internet technology based open-source power system simulation software development project, and discusses its system architecture, development process, extension mechanism and testing process. InterPSS features an open architecture, allowing advanced user to plugin additional modules to augment its functionality and/or change its behavior. InterPSS could potentially become an Internet-based platform for user participation, collaboration and further innovation in the area of power system simulation.**

Keywords: Power System Simulation, Open-source, Internet

I. Introduction

Over the last 10 years, the Internet has become the most far-reaching and extensive medium for exchange information and collaboration. Internet based economy has enjoyed explosive growth. Large corporations around the world have been quietly using the Internet and related technologies, such Java and Xml, to transform their information infrastructure to be more competitive and agile. The Internet technology has been used in solving a tradition power-engineering problem – power system simulation software development. InterPSS - an Internet technology based open-source power system simulation software system has been developed using the Internet as a virtual development platform and released under an open source license by the InterPSS development team.

It is our observation that power system simulation programs used in production today are largely written in procedure languages, such Fortran or C. It has been known that software systems developed using such approaches are expensive to maintain, difficult to be extended and hard to be integrated into other systems. It is our belief that sooner or later these programs will become obsolete and be replaced by new systems developed using modern software technologies - object-oriented programming languages and component-based development approaches.

Adapting new information technology to power systems has been relative slow as compared to other industries, such banking and telecommunication. We believe that the open-source approach with participation by universities, research institutes and power companies could be a promising way to speed up the process. Over the past few years, we have seen several university based open-source projects in power system simulation area, such as the Power System Analysis Toolbox (PSAT)[1].

M. Zhou is with TIBCO Software, Inc., Palo Alto, CA 94304 USA (e-mail: mike@interpss.org).

S. Zhou is with Thought Technology, Inc, Wuhan, Hubei, China (e-mail: zsz2004@gmail.com).

M. Zhou and S. Zhou represent www.interpss.org.

II. Project Overview

InterPSS is an open-source development project aimed to develop an Internet technology based software system for design, analysis, and simulation of power systems and targeted for real-world engineering usage. It features an open and loosely coupled plugin architecture, which allows components developed by others to be easily plugged into the system to augment its functionality and/or change its behavior, and equally important, allows its components to be integrated into other systems to provide certain power system simulation services.

InterPSS currently has two editions: Desktop edition and Web edition.

2.1 Desktop Edition

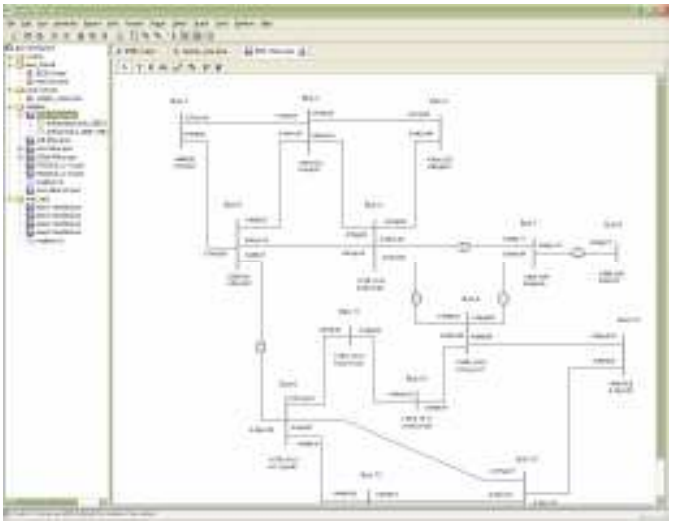

Fig.1 InterPSS Graphic Editor

InterPSS desktop edition features a graphic editor, as shown in Fig.1. It has a workspace with a project explorer where multiple projects could be managed. User can work on several projects in parallel, for example, editing a loadflow study project while performing a transient stability simulation case, which may take a long time to run. User can create a project by drawing a power system one-line diagram and entering data while drawing. After an analysis run, for example a loadflow calculation, the results (Bus voltage, branch P+jQ) could be annotated onto the one-line diagram. The editor also has an integrated reporting system, where user can generate different types of analysis reports on-demand.

2.2 Web Edition

Fig.2 InterPSS Web Edition

InterPSS started as a Web application in Chinese, as shown in Fig.2, hosted at home.interpss.org. From a Web browser, user can create a loadflow or short circuit project, or upload a data file in certain format, such as IEEE Common Format, to the site. Loadflow calculation and short circuit analysis can be then performed on the server. The results are posted back to the user browser screen.

The project is currently being developed and maintained by a group of volunteers living in China, Canada and the United States. The first version (Release 1.0) was released on September 17, 2006 under an open-source license GPL 2.0. There were more than 600 downloads in the first two months of its release.

III. InterPSS System Architecture

InterPSS is designed to be open, flexible and extensible with a plugin architecture. From the very beginning, our assumption is that InterPSS will be extended by others to do things that we might never have imagined. In this architecture, things which are common to power system simulation and less likely to change are packaged into the power system simulation framework as the InterPSS core library. The rest are implemented as plugins, which could be customized or replaced by user.

Fig.3 shows a high-level view of the InterPSS architecture. At the center is the power system simulation framework, which is based on the object-oriented power system simulation model proposed in Ref.[2]. Simulation algorithms, user interfaces, input/output modules, and integration interface implementations are plugins. These components and plugins are configured and integrated together by using the Spring Framework (www.springframework.org). The project team uses Eclipse IDE and NetBeans IDE to do their development.

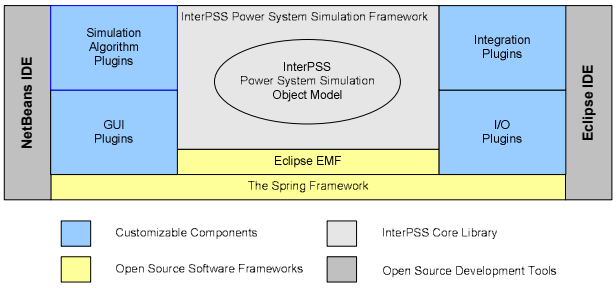

Fig.3 High-level Architecture View

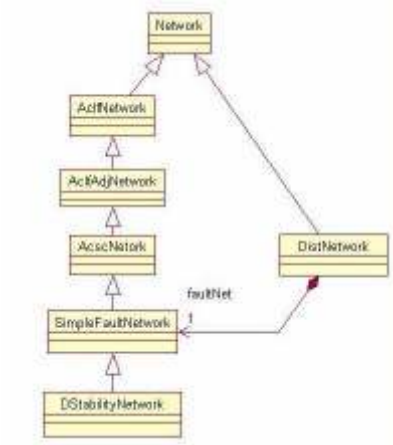

Fig.4 InterPSS Network Class Hierarchy

InterPSS uses the object-oriented approach[2] to model power system for simulation purpose. Fig.4 shows the current InterPSS network class hierarchy, where DStabilityNetwork for transient stability simulation inherits loadflow functionality from the AclfNetwork and AclfAdjNetwork classes, and short circuit analysis functionality from the AcscNetwork and SimpleFaultNetwork classes. DistNetwork is for distribution system analysis. Most traditional power system simulation software packages in the market today are consist of multiple programs, such as Loadflow program, Short Circuit program. Files or database are used for data exchange between these programs. InterPSS only has ONE runtime instance (program). Loadflow and shout circuit functionality is reused by the transient stability simulation module through object inheritance.

The project uses Java as its programming language. Java was born for the Internet and networked computers. It is especially suitable for building loosely coupled complex software systems. The Eclipse project (www.eclipse.org), which is the world largest, the most innovative and successful open-source project with hundreds of participating companies and millions of users, is developed in Java.

A common question about Java is its performance when it is used for power system simulation. Early versions of Java were significantly outperformed by statically compiled languages such as Fortran, C or C++. However, advances in Java Just-In-Time (JIT) compiler technology for long-running server and desktop Java processes has closed the performance gap and in some cases given the performance advantage to

Java[5]. In effect, Java byte code is compiled into machine instructions at run time, in a similar manner to C++ static compilation, resulting in similar instruction sequences. Further more, modern Java compiler tries to learn an application's runtime behavior and optimize the byte code compilation process for speed or memory usage. Early feedback from the user indicates that InterPSS is as faster as other power system simulation packages.

Xml are used to configure InterPSS components, for information exchange between these components, and for persisting information to database for late retrieval.

IV. InterPSS Development Process

The project currently has already become a rather complex software system with many modules/components/plugins. Managing its complexity is quite challenging, considering the fact that the project is concurrently developed by a group of developers, all volunteers, living in different countries, at their spare time. The project team uses the Internet as a virtual development workplace. The project takes full advantage of the following services and technologies.

4.1 Google Services

InterPSS uses Google services to manage its development process. The project source code is currently hosted at Google Code Service. The project website and email system is based on Google Office Service. The project uses Google Document service for its documentation and Google Spreadsheet for user testing process reporting and management.

4.2 Java IDE

The development team uses the Eclipse IDE (www.eclipse.org) and the NetBeans IDE (www.netbeans.org), both open-source IDE, for their team oriented development. Both IDE has advanced software development features, such as code re-factoring, integrated version control, debugging, profiling and unit testing.

4.3 Open-source Project Integration

The project makes heavy use of other open-source projects. Our guideline is that we will only write something if we cannot find it in the open-source world. Currently the project uses the following open-source packages:

- *Sprint Framework* (www.springframework.org) - A full-stack Java/J2EE application framework. It delivers significant benefits for Java development projects, reducing development effort and costs while improving test coverage and quality.
- *JGraph* (www.jgraph.com) - A Java graph drawing and layout component library. InterPSS one-line diagram is based on JGraph.
- *JasperReport* (jasperforge.org/) – A Java reporting tool that has the ability to deliver rich content onto the screen, to the printer or into PDF, Html, XLS, CSV and Xml files.
- *JFreeChart* (www.jfree.org/jfreechart) – A Java chart library that makes it easy for developers to display quality charts in their applications.
- *Castor* (castor.codehaus.org) - A data binding framework provides Java-to-Xml binding.
- *Apache Derby Database* (db.apache.org/derby) - A relational database implemented entirely in Java.
- *Apache iBATIS* (ibatis.apache.org) – A data mapper framework, coupling objects with stored procedures or SQL statements using a Xml descriptor to make easier to use a database with Java applications.
- *Apache Commons* (www.apache.org/commons) – A set of Java libraries for logging and complex number.
- *JUnit* (www.junit.org) – JUnit is a regression testing framework. It is used by millions of developers who implements unit tests in Java.
- *FitNesse (www.fitnesses.org)* - FitNesse is a software development collaboration tool for enhancing collaboration in software development.
- *Eclipse/EMF (*www.eclipse.org/emf*)* – See Section-4.4.

4.4 Code Generation

The majority of InterPSS core library code is generated by using an open-source technology, called Eclipse EMF (www.eclipse.org/emf), which is a modeling framework and code generation facility for building tools and other applications based on a structured data model. Shown in Fig.5 is a UML diagram describing a simplified power system network topology model[2]. Eclipse EMF is used to translate the model into Java classes, which are highly efficient based on our benchmark.

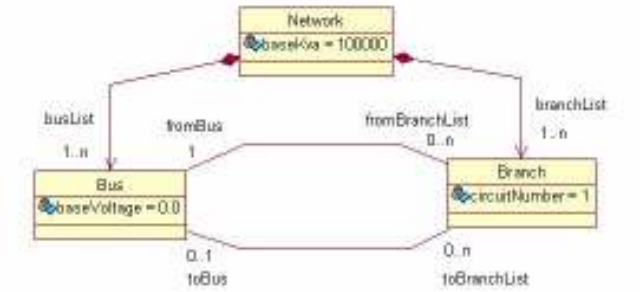

Fig.5 A Simplified Power Network Model

InterPSS is an Internet technology based open-source project. It is totally depends on the Internet infrastructure, Internet services provided by other companies, such as Google and Yahoo, and other open-source projects. InterPSS project would not have been possible without the Internet and the open-source movement.

V. InterPSS Extension

InterPSS, from the beginning, was designed to be open and extensible. The InterPSS team had realized that their resource is limited. It is not feasible to implement and maintain all possible power system simulation functions and algorithms. The IntePSS plugin extension architecture is our solution, which allows any one to extend InterPSS functionality and/or change its behavior by developing plugins.

The following is a simple example to extend InterPSS. The original question came from a University in China. They would like to measure InterPSS loadflow computation time on different operating systems, Windows vs Linux. InterPSS core library does not log timestamps suitable for this kind research work. However, by writing few lines of code, InterPSS could be extended to do the measurement. The following outlines the maintain steps:

- Implement a new loadflow algorithm, as follows:

```
/*
* LoadflowAlgorithmImpl is InterPSS default
* loadflow implementation
*/
public class MyLoadflowAlgorithmImpl
             extends LoadflowAlgorithmImpl {
   public boolean loadflow(IPSSMsgHub msg) {
      Date start = new Date();
      // super here means using InterPSS default
      // loadflow algorithm
      boolean status = super.loadflow(msg);
      Date end = new Date();
      // print out the time difference for
      // time consumption measurement.
      return status;
   }
}
```

- Modify one of InterPSS configuration files to plug-in the new algorithm:

```
<bean id="loadflowAlgorithm"
   class=<full classpath of the implementation>
   singleton="false">
</bean>
```

The above extension is quite trivial. It is unlike that you will use it to solve real-life problems. However, the extension concept is very powerful.

- InterPSS already has basic power system simulation functions and algorithms, such as sparse equation solution, loadflow, short circuit and transient stability simulation. InterPSS may not behave exactly as you expected, for example, no easy way to calculate loadflow CPU time consumption. By the extension mechanism, you can easily modify InterPSS's behavior.
- In the extension process, you do not write code from scratch. You reuse InterPSS's existing functions, which have been fully tested, and add your personal touches. In this way, you will be able to come up with a solution much faster as compared to writing your own solution from scratch.

Using extension approaches to solve power system simulation problem is not new. For example, PTI PSS/E uses an extension approach in its transient stability implementation, see Ref[3] (page 11-8, Fig. 11-6) for details. However, InterPSS uses modern object-oriented approach and the Inversion of Control[4] software design pattern, as compared with PSS/E's Fortran style approach. It is our belief that the InterPSS approach is more robust, flexible and maintainable.

VI. Community-based Testing Process

History has shown that a development team alone cannot ensure the quality of the software they developed. It should be tested by a separate team and the end user. During InterPSS development process, JUnit (www.junit.org) is used for unit testing. Hundreds of unit test cases are built to make ensures that individual software components produce expected results. JUnit-based regression testing is performed to verify that software components work together correctly after changes are made. InterPSS JUnit testing code is open and can be examined by any interested user.

InterPSS also has a FitNesse (www.fitnesse.org) based Wiki style user acceptance testing site. User can examine and run the test cases to perform the official InterPSS user acceptance testing at any time. User can also build his/her own test cases on the site. User can upload test data files and user-developed plugins to the site to build his/her test cases. The user-defined test cases could be run repeatedly against any future project release.

While the project currently has hundreds of automated unit and user acceptance test cases with more to be added, InterPSS also relies on human to perform functional testing using more complex use cases. The test procedure is open. Test results are published in a on-line Google Spreadsheet report, which could be viewed by user and updated by testing participants.

The InterPSS team knows that GOD is the only person who can write bug-free code. As a human being, we always make mistakes. A user community-based testing and quality assurance (QA) process is currently under development, so that the user community can watch the development and participating in control the software quality before each new release. The FitNesse based InterPSS user acceptance testing site enables users, testers, and programmers to learn what the software should do, and to automatically compare that to what it actually does do. It compares user's expectations to actual results. It will serve as the foundation for the upcoming InterPSS user community based testing and QA process.

VII. Call for Participation and Contribution

A fully functional power system simulation software system has been developed, and offered to the power engineering community under an open-source license (free download and usage). The project has the potential to serve as a platform for user participation, collaboration and further innovation. We believe that InterPSS's future depends on the participation and contribution from the user community. We hope that a strong synergy could be developed between the project team, power engineers and researchers from universities, research institutes and power companies. Therefore, we are calling for participation and contribution!

We believe everyone can participate and contribute. We envision that the user community will consist of the following groups, with the largest number in the regular user category, as shown in Fig.6:

- Regular User – Use software regularly or occasionally for certain purpose, and feedback issues and suggestions.
- Participating User – In addition to use the software, actively participate in the community discussion, software testing and QA process, and contribute to the project Wiki to share experience and knowledge.
- Contributor – Develop plugins or add-ons to augment the functionality of the software and contribute back to the user community.
- Committer – Work on project plugin(s) and/or subsystem(s) on an on-going basis as a leader developer or project coordinator.
- Architect – Help defining the future direction, including prototyping new technologies and investigating their applicability to power system simulation.

The project also provides a Wiki site, as shown in Fig.7, where technical information, user guide, reference manual and future development roadmap are published. The Wiki site is open to the user. User can participate and contribute contents to the Wiki site to share user's experience and knowledge.

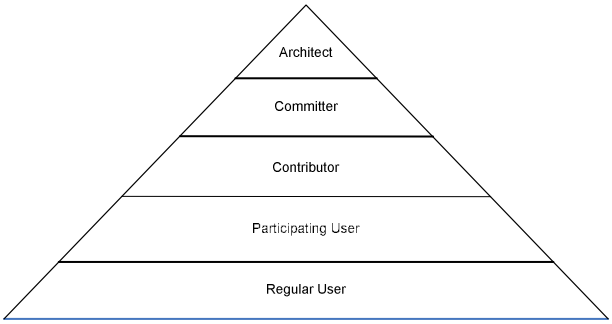

Fig.6 User Community Structure

Fig. 7 InterPSS Wiki Site

VIII. Summary

An Internet technology based power system simulation software system - IntePSS has been developed and released under an open-source license. It features an open extensible plugin architecture, where its functionality and behavior could be augmented and/or changed by advanced user. Because of its open nature, the project has the potential to serve as a platform for user participation, collaboration and further innovation. InterPSS depends on the Internet infrastructure, Internet services provided by other companies, such as Google and Yahoo, and other open-source projects. The InterPSS development team hopes that others will join the open-source effort to make InterPSS a mainstream power system simulation software system in the near future.

IX. References

[1] F. Milano, "An Open Source Power System Analysis Toolbox", IEEE Trans. on Power Systems, Vol. 20, No. 3, August 2005, pp. 1199-1206.

[2] E.Z. Zhou, "Object-oriented Programming, C++ and Power System Simulation", IEEE Trans. on Power Systems, Vol.11, No.1, Feb. 1996 pp206-216.

[3] "PSS/E™ 29 – Program Application Guide Vol.II", Power Technologies, Inc, Oct 2002.

[4] Martin Fowler, "Inversion of Control Containers and the Dependency Injection pattern", http://www.martinfowler.com/articles/injection.html.

[5]"Comparison of Java and C++", Wikipedia, http://en.wikipedia.org/wiki/Comparison_of_Java_and_C++

X. Acknowledgement

The authors would like to take this opportunity to acknowledge late-Prof Wang Zonggan, Prof Chen Shousun, late-Prof Zhang Baolin and Prof Zhou Rongguan of Tsinghua University, Beijing China for their dedication in power system engineering research and education.

XI. Biographies

Michael Zhou, Ph.D. Tsinghua University, senior computer system architect with TIBCO Software, Inc., Palo Alto, CA. He is the architect of the InterPSS project. He is also an adjunct professor, College of Electrical Engineering, Hunan University, Hunan, China. Ref.[2] was written by the author.

Shizhao Zhou, B.Sc., Wuhan University of Technology, computer software system architect with Thought Technology, Inc, Wuhan, Hubei, China. He is the leader developer of the InterPSS graphic editor.