DEVELOPMENT OF THREE PHASE LOAD FLOW BY REUSING OBJECT ORIENTED SINGLE PHASE LOAD FLOW

Mamdouh Abdel-Akher, Khalid Mohamed Nor, Hazlie Mokhlis and Abdul Halim Abdul Rashid
University of Malaya, Malaysia

ABSTRACT

This paper presents the development of three phase load flow software by reusing an existing single phase load flow software component. The single phase load flow component was developed using Object Oriented and Component Based Development methodologies. The object oriented power system model of the single phase load flow component was established separately from the mathematical solution. The proposed three phase load flow is formulated based on symmetrical components theory, which involves a positive sequence load flow solution. Based on this relation, the single phase load flow component and its power system model can be reused in the development of the three phase load flow software. The single phase load flow power system model is extended for modeling three phase devices such as untransposed lines, three phase loads... etc. Reuse of the single phase component is achieved through composition relationship, where the positive sequence load flow is modeled as an object inside the three phase load flow component. The three phase load flow is solved by incorporating the solution of both positive sequence load flow and the nodal voltage equations for zero and negative sequence networks.

Keywords: load flow, Software Reuse, Object oriented, Symmetrical components, Newton-Raphson, Fast-Decoupled.

INTRODUCTION

Three phase load flow analysis is very important in electrical power system that considering three phase unbalanced condition. Power system planning, operation, control and optimization are few examples of it wide application. The three-phase load flow study can be achieved in phase coordinates or in symmetrical components frame of reference. Most of studies are established in a-b-c frame of reference. This is because the mutual inductances between different phases of an unsymmetrical transmission lines is not equal to each other. Recently the decoupled models of the untransposed transmission lines were presented [1]. These models are used for solving three phase load flow based on symmetrical components theory [1]. These techniques are implemented in this research for developing new integrated load flow simulation using component object model (COM) and object oriented programming methodology (OOP).

Before the introduction of OOP, software were built using structural programming approach. However, this approach has a lot of drawbacks such as has high development cost, low productivity, unmanageable software quality, and high risk to move to new technology [2]. The OOP is then came to resolve this problems. OOP is a natural extension of structured programming. It provides a good programming practices and makes it very easy to do so. The result is clean code that it is easy to extend and simple to maintain. Once an object is created for an application, the same object can then be used in other applications. Reusing objects can greatly cut the software development time and increase productivity [3]. An object represents an individual, identifiable item, unit, or entity, either real or abstract, with a well-defined role in the problem domain. It has state, behavior, and identity; the structure and behavior of similar objects are defined in their base class; the terms instance and object are interchangeable. The objects are classified into three types. Firstly, Entity Objects that represent objects in our real world. Secondly, Interface Objects that handle data exchanged with users or other systems. Finally, Control Objects that are created to handle complex operations that do not fit naturally in any one object or class [4].

Although OOP provides reusability features, this is only at the source codes level since the codes still have to be recompiled whenever their functionalities need to be extended. In order to overcome this limitation, a newest software development approach that is Component Based Development (CBD) can be used. This approach is very different from traditional approach in which software systems can be implemented from scratch. The components can be developed by different software developers in different times and at different places. Components give us a head start for developing our software by maximizing mechanisms such as components reusability. There is no existing standard definition of the key item "component". In general a component has three main features [5]:

- It is an independent and replaceable part of a system that fulfills a clear function.
- It works in the context of a well-defined architecture.
- It communicates by its interfaces.

In this paper, the power system models that present power system devices as well as solution algorithms are encapsulated inside objects. The power system devices are modeled as entity objects. These entity objects present the object-oriented power system model (OO-PSM). On the other hand, the solution algorithms such as Newton Raphson, Fast Decoupled, and Linear Solver are modeled as control objects. These control objects contain complex solution algorithm that can not be fit in a single class. The control objects are developed as independent components. These control objects present the power system component object model (PS-COM).

SINGLE PHASE OBJECT ORIENTED POWER SYSTEM MODEL

The real power system devices, such as generators, transformers, transmission lines, and loads are modeled as objects and classified by the same way as in the real world. Fig. 1 shows the class hierarchy of the power system models. These models are designed to be member data of the components that are developed to solve the load flow. So, these models are reused through out the different power system studies because they became data type for the power system.

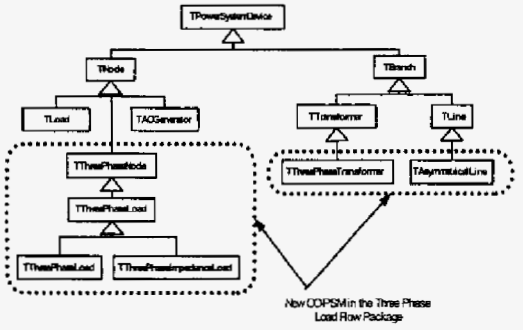

Figure 1 The Object Oriented Power System Model

In Fig. 1, the electrical network is described by two essential elements. These elements are nodes and branches. They are modeled as classes TNode and TBranch. The elements that are connected to nodes such as generators and loads are inherited from the class TNode. On the other hand, the elements that are connected between two busbars such as transmission lines and transformers are inherited from the class TBranch.

DESCRIPTION OF THE SINGLE PHASE LOAD FLOW COMPONENTS

The single phase load flow contains four components; each component has a clear function in the single phase load flow simulation. There are three components appears in Fig. 2, TFDecoupled, TNRaphson, and TLinearSolver. The fourth component is an interface component for reading the IEEE data format, this component is used in CBD applications and does not appear in Fig 2.

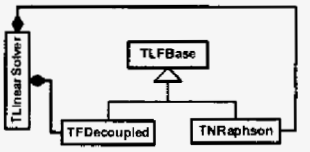

Figure 2 Single Phase Load Flow Component Software

Single Phase Load Flow Base Class

This class is designed to be base class for any single phase load flow algorithm. The class contains the common data, methods, and interface methods based on the single phase OO-PSM. Any algorithm for solving single phase load flow can be encapsulated inside a component inherited from the base class. The protected part of the class contains the OO-PSM. The OO-PSM is visible for all the class descendants. An interface for the OO-PSM is established to enable the user form accessing the OO-PSM. To make the components reusable, two important flag interfaces are included in the class. These interfaces control the behavior of the inherited components from the class. The first flag interface enables the user to perform the solution process for one iteration and at the same time the refactorization is avoided in the next iteration. The second flag interface checks the convergence inside the component.

The Linear Solver Component

The solution of the load flow requires a mathematical solver for handling the matrix operation involved in the solution process. The solver component uses SuperLU library routines [6]. The solver can solves matrix equation in the form of $Ax=b$ in order to get the x values. Although it is a public domain code, it still has the same capability compared to other in solving sparse linear equation. It uses many latest techniques, such as graph reduction technique in matrix factorization. Furthermore, it can also solve very unsymmetrical matrices.

Single Phase Load Flow Components

The function of these components, TNRaphson and TFDecoupled, is to solve the load flow using both Newton Raphson and Fast Decoupled algorithms respectively. The mathematical solver component is aggregated in both the two components for handling the sparse matrix operations accompanied with the solution process as shown in Fig 2. The interface component is developed to read the standard IEEE data format. The component encapsulates all the power system data in one interface object. The component can be reused through

out different CBD applications for different power system studies.

EXTENDING OO-PSM FOR MODELING THREE PHASE DEVICES

The three phase load flow components require three phase OO-PSM to represent the real three phase elements such as three phase busbars, three phase loads, three phase transformers ...etc. In order to have the required model, the OO-PSM of the single phase in Fig. 1 can be extended for three phase OO-PSM by inheriting the single phase OO-PSM.

DESCRIPTION OF THE THREE PHASE LOAD FLOW COMPONENTS

The three-phase load flow solution algorithm (Positive Sequence Load flow and Nodal Voltage Equations) is encapsulated inside components independent on the algorithms used in the single phase load flow components. The single phase load flow components are reused based on composition only. This reduces the dependency between the codes of both single phase and three phase load flow components.

Three Phase Load Flow Base Class

The class is designed to be the base class for any three phase load flow algorithm uses the symmetrical components theory and the decoupled models of the untransposed transmission lines. The class is developed based on object oriented approach; it handles the single phase load flow as an object. This object behaves as black box, only communicates with its surroundings with its interface. The flag interface in single phase load flow base class, is used for controlling the behavior of the component. The first one is adjusted such the positive load flow object performs one iteration while the second checks the mismatch tolerance is acceptable or not. The object returns the positive sequence load flow solution at each iteration during the overall solution process.

The three phase load flow is solved by incorporating the positive load flow solution with both the solution of the zero- and negative- sequence nodal voltage equations. The linear solver component is aggregated in the three phase load flow base class for solving the both zero and negative sequence voltage equations as exhibited in Fig. 2. The positive sequence load flow object can be declared as any type of the single phase load flow algorithms. It may be TNRaphson or TFDecoupled. According to its declaration, the positive sequence object will behave in three phase load flow base class. The declaration of the positive sequence load flow is done inside independent components inherited form the three phase load flow base as exhibited in Fig 3.

Three Phase Load Flow Components

There are two components for solving three phase load flow as given in Fig. 2. The two components are inherited from the base class TThreePhaseLoadFlow. The first component TSNRaphson declares the positive sequence load flow object as TNRaphson. The second component, TSFDecoupled, declares the positive sequence load flow object as TFDecoupled. Fig. 2 presents the component object model for the load flow analysis software.

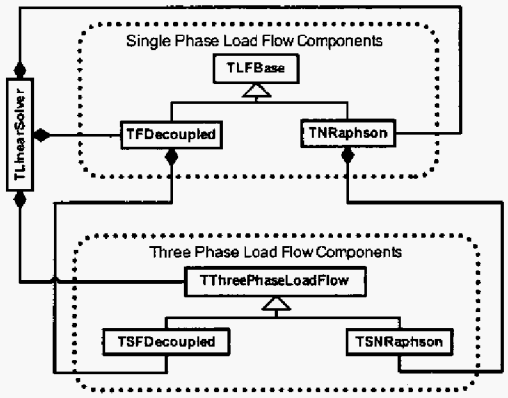

Figure 3 Load Flow Object Components Model

The most important point that makes these components flexible, reusable, and can be replaceable that the interface of the load flow components are based on the OO-PSM model itself, this model is persistent, i.e. long life cycle object. Normally these objects are not related to algorithms that are encapsulated inside load flow components. The OO-PSM is only reused as data type for developing the load flow components.

Interface Component For Unbalanced Three Phase Data

This component is developed as interface component; the components collects all the unbalanced power system data inside one interface object, these data include the unbalanced loads, untransposed transmission lines, the sequence data for transformers, generators ...etc. The component is not shown in Fig. 3. It is reused only in CBD application for supplying the power system data for the components that request these data.

PACKAGING COMPONENTS

A package is a special dynamic-link library used by C++Builder applications [3]. The load flow object component model and OO-PSM can be packaged inside

one or more than one package. Then, the package is compiled to generate the package libraries. The libraries are stored in files that have extension with the bpl (Borland package library). The library then can be reused for developing many functioning application for both single and three phase load flow or can be extended for developing new software components that could perform another power system study.

COMPONENT BASED DEVELOPMENT APPLICATIONS

The source code of the developed components and classes are not required, since the library files plus the header files only required for developing functioning applications. Two CBD applications are given below to explain the usage of the object load flow components.

Component Based Development Single Phase Load Flow Application

Table 1 gives the code for the single phase load flow application. Firstly, an interface object called IEEEData is declared, this object has a job that is to read the power system data in the standard IEEE data format. Then control object, LoadFlow is defined as instant from the class TLFBase. There are two possibilities for declaring the object LoadFlow. The first choice is as the TNRaphson component while the second as the TFDecoupled component. The power system data is transferred form the interface object IEEEData to the object LoadFlow by its interface.

Table 1 Single Phase Load Flow CBD Application

```
#include NRaphson.h
#include FDecouple.h
#include ReadAsciiData.h
:
TReadAsciiData *IEEEData=new TReadAsciiData(NULL)
IEEEData ->FileName(IEEE300.txt)
:
TLFBase *LoadFlow;
if (YourChoice==NewtonRaphson)
LoadFlow=new TNRaphson(NULL);
if(YourChoice==FastDecoupled)
LoadFlow=new TFDecouple(NULL);
LoadFlow->AddBus( IEEEData ->NBus, IEEEData ->Type.....)
:
LoadFlow->Calculate();
LaodFlow->PrintResult();
```

The load flow is solved by dispatching the interface method Calculate(). The method is defined inside the class TLFBase as virtual method. The method will be invoked at the runtime according to the declaration of the object LoadFlow. If the object is declared as TNRaphson, the method Calculate() inside the component TNRaphson will override the virtual method inside the base class TLFBase. The same situation if the control object LoadFlow is declared as TFDecoupled. The solution is printed in text file by dispatching the interface method PrintResult().

Component-Based Development Three Phase Load Flow Application

The three phase load flow components are used the same way as in the case of the single phase load flow application. Addition data should be supplied to the object LoadFlow in Table 2. This data is encapsulated inside the interface object ThreePhaseData. Three phase load flow in the application can be solved by dispatching the interface method Calculate() in the object LoadFlow in similar manner to the case of the single phase load flow application.

Table 2 Three Phase Load Flow CBD Application

```
#include SNRaphson.h
#include SFDecouple.h
#include ReadAsciiData.h
#include UnbalncedData.h
:
TReadAsciiData *IEEEData=new TReadAsciiData(NULL)
TUnbalancedData *ThreePhaseData=new
TUnbalancedData(NULL);
IEEEData ->FileName(IEEE14.txt)
ThreePhaseData->LoadFileName(...);
ThreePhaseData->LineFileName(...);
:
TAsymmetricalLoadFlowLFBase *LoadFlow;
if (YourChoice==SequenceNewtonRaphson)
LoadFlow=new TSNRaphson(NULL);
if(YourChoice==SequenceFastDecoupled)
LoadFlow=new TSFDecouple(NULL);
LoadFlow->PositiveLoadFlow->AddBus( IEEEData ->NBus,...)
:
LoadFlow->Calculate();
LaodFlow->PrintThreeResult();
```

REUSABILITY IN THE DEVELOPMENT OF THE THREE PHASE LOAD FLOW

The reusability is measured in software industry for economic reasons especially in developing large software projects. In this paper component technology is used for developing reusable load flow objects. These objects model complex problems that may take along time to be developed from scratch. The single phase load flow object components are reused for developing other components for three phase load flow. And at the same time the OO-PSM is extended to new models.

Table 3 presents the developed components from scratch and the reused components from single phase. Simple measurement for the reusability can be done by counting both the classes and components that are developed from scratch; those are reused based on composition, and those are modified by inheritance form the code in the single phase load flow. The result exhibits that 55% of the three-phase OO-PSM is developed by inheriting from the

single phase OO-PSM. The rest 45% is reused as it in the development of the three phase load flow based on composition. In view of the solution algorithms, 60% of three phase components are reused based on composition of single phase load flow components and the rest 40% are developed from scratch.

Table 3 Reusability in Developing 3-φ Load Flow

Reusability	Class		Component	
	Total Number	Percentage %	Total Number	Percentage %
Composition	5	45	3	60
Inheritance	6	55	-	-
Scratch	-	-	2	40
Total	11	100	5	100

It is important to be mentioned here in that to produce reusable component, this requires understanding for the problem that the object component represents. This means that developing reusable component for power system study is not only programming practice but also power engineering practice.

RESULTS

The 24 bus data in reference [7] was tested with both the two CBD application in section 7. In the case of both power system and load are balanced, the results differs only in the phase shift introduced by Y-Δ transformers. For unbalanced power system, Case B in [7] was tested for sake of comparison, the sample results in Table 4 agree well with those given in Table 5 in [7].

Table 4 Results for Three Phase Load Flow

Bus	Type	Phase A		Phase B		Phase C	
14	0	0.993	19.5	0.998	-100.5	0.995	139.2
32	0	0.983	11.6	0.994	-108.0	0.995	131.3
41	0	0.957	6.5	0.969	-112.9	0.973	126.1
45	0	0.973	5.5	0.983	-113.9	0.988	125.2
70	0	1.005	22.3	1.010	-97.8	1.008	142.0
71	0	0.991	14.3	1.002	-105.3	1.003	134.0
77	0	0.982	8.9	0.995	-110.5	0.997	128.5
97	0	1.000	20.4	1.004	-99.8	1.001	140.1
113	2	1.030	0.6	1.031	-119.5	1.029	120.5
114	0	1.016	24.9	1.019	-95.3	1.015	144.7
120	3	1.030	0.1	1.031	-120.0	1.029	120.0
134	0	0.989	9.5	1.002	-110.0	1.003	129.2
181	2	1.030	-1.8	1.031	-121.9	1.029	118.1
190	0	0.978	14.9	0.984	-105.0	0.984	134.6
193	2	1.045	-13.5	1.052	-133.3	1.053	106.3
196	0	0.996	10.2	1.007	-109.4	1.008	129.9
312	2	1.030	-1.8	1.031	-121.9	1.029	118.1
Total Load				3620.00 MW		650.00 MVA	
Total Generation				3651.89 MW		856.45 MVA	

CONCLUSIONS

The paper has presented the development of three phase load flow reusing single phase load flow based on symmetrical components theory. There are two object-models were presented in this paper. The first one models the power system devices, OO-PSM. In this model, the single phase model is extended for modeling the three phase devices. The second one is the load flow component object model that models the load flow solution algorithms single phase or three-phases. In this model, the single phase load flow objects are reused for developing the three phase load flow components. Both of single phase and three phase load flow objects are reused for developing CBD application. These applications were tested, only sample results for three phase load flow were presented.

ACKNOWLEDGMENT

The authors gratefully acknowledge University of Malaya in the work reported in this paper

REFERENCES

1. Xiao-Ping Zhang, "Fast three phase load flow methods" Power Systems, IEEE Trans. on Power System, Vol. 11, No. 3, Aug 1996

2. Developer's Guide, Borland C++Builder 6 for windows.

3. Xia Cai, Maichel R. Lyu, Kam-Fai Wong, and Roy Ko, "Component-Based Software Engineering: Technologies..." Proc. APSEC2000, 2000.

4. Ivar Jacobson, "Object-Oriented Software Engineering" Addison-Wesley Publishing Company. 1992.

5. Luiz Fernando Capretz, Miriam A, M Capertz, and Dahai Li, "Component-Based Software Development", Proc. Of IEEE IECON'01, 2001.

6. J. W Demmel et. al, "A Supernodal Approach to Sparse Partial Pivoting", SIAM Journal on Matrix Analysis and Applications Vol 20 , No 3, pp.720 -755, 1999.

7. Chen, B.-K. Chen, M.-S. Shoults, R.R. Liang, C.-C., "Hybrid three phase load flow", IEE, Generation, Transmission and Distribution, Vol 137, No 3, May 1990, pp 177 -185.

AUTHOR'S ADDRESS
The second author can be contacted at:
Department of Electrical Engineering,
University of Malaya
50603 Kuala Lumpur, Malaysia
Email: khalid@um.edu.my