AN OBJECT ORIENTED POWER SYSTEM SIMULATOR

M. Belkacemi, S. A. Alghamdi
College of Technology at Al Baha
PO BOX 6, Al Baha, Kingdom of Saudi Arabia
E-mail: belkacemi_m@hotmail.com

L. Benfarhi
Department of Electrical Engineering
University of Batna, Batna 05000, Algeria

ABSTRACT

In order to provide aids to the power system dispatchers and electrical engineers, a prototype package has been developed for power system planning, analysis, design and education. The chief advantage of this package is featured by its capability to combine texts and graphics into one package. It contains several options such as extensive color graphics representations of power systems, interface with external power system simulation programs, dynamic display of the simulation results and communication among external and internal data bases. Through interactive and friendly graphic interface, the package provides a guideline for the user to understand how the power system operates. The paper illustrates application of Object Oriented Technology in the development of a power system simulator named SHABAKA. With this technique, a power system object model is designed based on the real world concepts and implemented with the object oriented programming technique. The software is capable of supporting a wide range of power sytstem applications and allowing modification and maintenance over a long period of time.

1. Introduction

During the last three decades, a large number of software systems were designed to provide various engineering analysis for the planning, design and operation of today's complex electrical power networks. In recent years, the rapid advances in micro-computer hardware technology have increased the availability of power application software on personal computers. These software do now exist for tasks ranging from Steady-State Analysis to On-line Supervisory Control and Data Acquisition. To improve the performance of the new generation of power system software, interactive and graphical facilities are now continuously introduced to meet the requirements of power system operation.

The main reason for the effectiveness of interactive computer graphics is the speed of interpretation of the results, and the easy communication between the user and the computer. Essentially, the pertinent data is often more easily understood when the information is represented by graphics compared to the pure numerical representation in tabular forms, where drawings, scanned images, color and even animation can further enhance the text expression and increase user productivity [1-3].

However, to meet the new requirements of power system operation and in order to introduce the new computer techniques, most of the power system software need to be upgraded to meet the following demands [4-6]:

1. Flexibility: The power system and the requirements change permanently in a deregulated environment.
2. Expandability: new functionality has to be built into existing programs and entirely new functions have to be integrated.
3. Maintenance: modifying a piece of code must not affect other parts of the program.
4. Data integrity: must be realised in an easy way and independent from any power system analysis application.

The consequence is that some of the traditional systems have become so complicated that the applied software design practices are inadequate to support further enhancement and maintenance. As a result, replacement of the entire system becomes inevitable. To overcome the above drawbacks of the existing software, Object Oriented Technique (OOT) has gained widespread acceptance in software engineering by implementing complex problems as its advantages of flexibility and ease of maintenance have been recognized [4-6].

This paper explores the prospect of applying this technology in the development of a power system simulator named SHABAKA. With this technique, a power system object model is designed based on the real world concepts and implemented with the object oriented programming technique using DELPHI programming language. SHABAKA is equipped with a graphics user interface (GUI) and a data base. It is capable of supporting a wide range of power system simulations ranging from Load Flow analysis to

Protection Coordination . The objective of the project can be summarized as a developme nt of an integrated software environment capable of supporting a wide range of power system applications and allowing expansion, modification and maintenance over a long period of time.
An overview of the Object Oriented Technology (OOT) will be first given then the simulator SHABAKA structure based on the object oriented programming techniques will be illustrated together with the incorporated graphical and numerical simulation facilities.

2. Overview of the Object Oriented Technology (OOT)

Object oriented technology is a new methodology of software development and is arguably one of newest and far reaching developments in the computer industry. Its greatest benefits come from helping developers express abstract concepts clearly and communicate them to each other.

In the object oriented approach, the decomposition of the problem is based upon the concept of an object. An object is an entity whose behavior is characterised by the actions that it suffers and that it requires other objects. Each module in the system denotes an object or class of objects from the problem space.

Objects are the software modules which incorporate instance variables (data) and a set of methods (operations) to operate on the data. Abstract and information hiding (encapsula tion) form the foundation of object oriented development.

Object oriented programming technique is very flexible because changes to one object will not affect other objects of the programs. Thus, it allows engineers to focus on applying the knowledge of their field rather than on computer related details such as keeping the variables and data structures of various functions consistent[6].

The object oriented developments as model based software developments are:

1. Conceptual design of objects in the application domain serves as basic approach.
2. Objects oriented models constitute a common discussion basis for systems analysts, developers and customers.
3. Object oriented modeling helps narrow the gap between application domain and software components.
4. Strength of this approach is the use of the same conceptual abstractions, notation and philosophy in every phase of the software life cycle:
   - Develop the analysis models.
   - Develop the design models from the analysis models.
   - Implement the design models.

2.1 Object oriented analysis

The purpose of the OOA is to state and understand the problem and application domain so that a current design can be constructed and the object model, the dynamic model and the functional model are build.

2.2 Object oriented design

During design stage, decisions were made about how the problem will be solved. This includes system design and object design. In the system design the many decisions need to be made such as how to:

1. Organize the system into subsystems,
2. Choose an approach for management of data stores.

The object classes and relationships developed during the object design are finally translated into a particular programming language. In this phase all the parts of the program were integrated and tested

3. The Structure of SHABAKA

The main feature of SHABAKA is the combination of the graphical and computation facilities. The Object Oriented Technology is used here with the purpose of improving the software flexibility and maintenance together with the use of other new programming techniques to reduce both memory space and computing time requirements.
The package is implemented using DELPHI programming language under Microsoft Windows 98. The advantages of windows based programming are fully explored including standard functions and options .
The overall structure of the simulator SHABAKA is illustrated in figure 1.

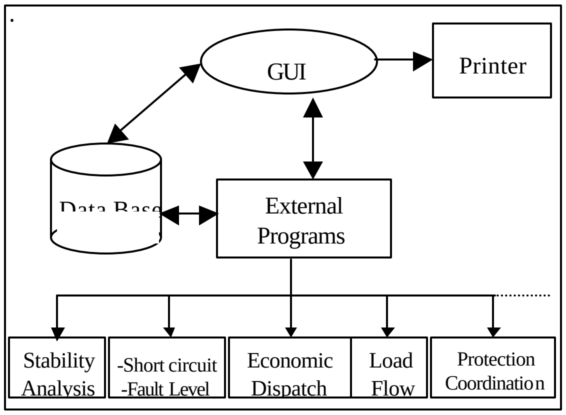


Figure1 The main components of **SHABAKA**

The designed power system software is composed of variety of application modules. Each of them is capable

of analyzing one aspect of power system operation and may require data input in different formats.
In order to meet the open system requirement, a distributed software architecture is adopted to configure the system. As shown in figure 1, each component is an independent module and interacts with o thers through message passing.

The power system computation modules are consisting of a set of external programs developed separately using different programming languages and they are linked to the Graphics User Interface (GUI). Once a power system file is constructed, by drawing a power system line diagram using the GUI and inputting its numerical data in the corresponding tables using the Data Base (DB) or by retrieving an already saved file. The user can choose from a menu to run one external program at a time such as Load Flow, Short Circuit , Economic Dispatch, Transient Stability or Fault Level

3.1 Graphics user interface (GUI)

The GUI allows users to interface with external programs, to display simulation results as well as the following data:

- Summary tables and system diagram of each subsystem
- Physical characteristic data
- Operational characteristic data

SHABAKA environment is equipped with a main menu, submenus with pull drawn bottoms. For example in the ″File″ menu, the user can open, save, close, print or quit the simulation session. In the″Calculate″ menu, the user can execute one of the simulation programs such as Load Flow, Fault Level or other.

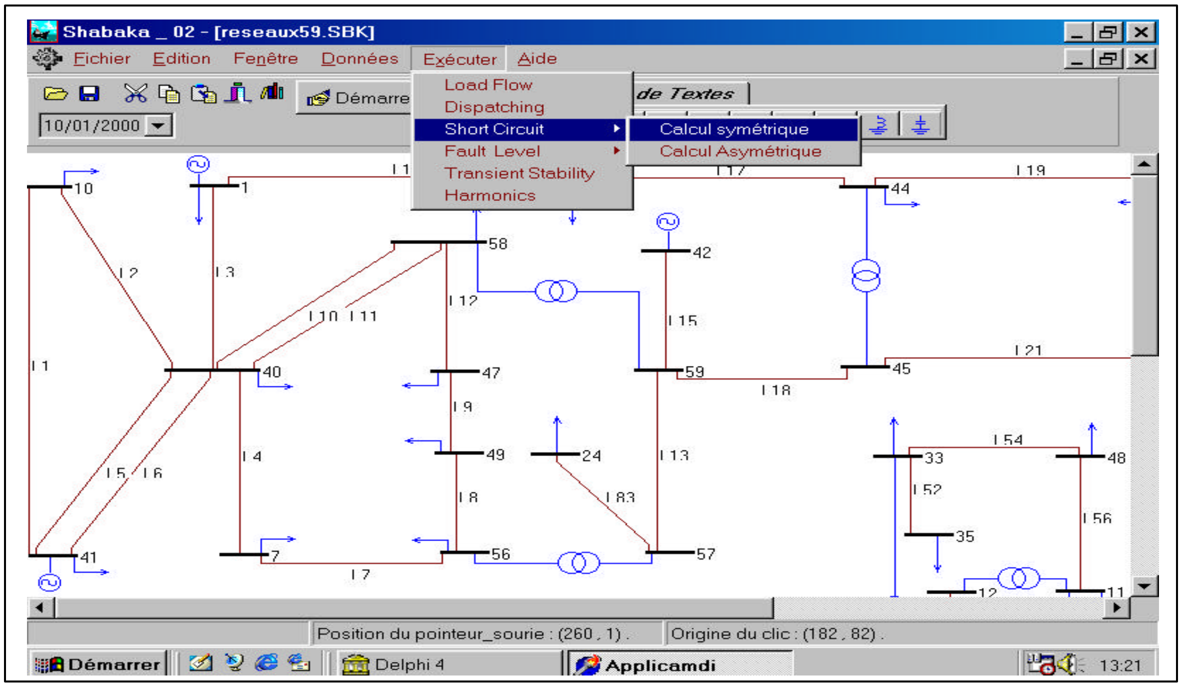


Figure 2: Algerian 89-Bus, 10 Generators System

Power system graphical components in GUI are created from basic graphical elements (lines, circles, rectangles, etc.). The GUI draws transmission lines, buses, loads, generators, motors, transformers, etc. These elements can be linked together to construct any desired power system configuration. Each component of a power system can be drawn in either a horizontal or a vertical position at any point on the screen. Figure 2 shows the line diagram of a real power system as displayed on the screen together with the list of the drawing and computing options available on the GUI menu.

In addition, GUI user can edit large blocks of information using basic commands such as cut, copy paste and clear. These functions allow user to modify the configuration of the study power system at any time by using the mouse and clicking the field which contains the graphics component or the line flow data to input the new values.

Drawing and modifying graphically a power system network is not a difficult task, but storing the numerical data describing its position, appearance, and relations is an important problem. A special program has been developed to solve this problem by saving both graphical and numerical data of any system.

3.2 Data base (DB)

Generally, manipulation of data is the most tedious and time consuming tasks in power system computation. SHABAKA addresses this problem by using data editors which are provided for each network element.

Tables are created according to the class name of the network elements . Several functions are developed to display on the screen contents of these tables and where the user can edit or modify the data easily.

There are two types of data :

1. The first is the data describing the network configuration and is used to define the table dimensions of the data and the data base structure. Direct modification of this data is not permitted.
2. The second type is the data for the study system that can be accessed and changed by the user.

Figures 3, and 4 are too examples of data management for a nine bus system. They show the variables which need to be introduced or changed.

Several functions have been developed in this module and are used to input, load or modify data in an interactive way. In addition to the necessary data describing the graphical representations of power systems, the following c lass tables are created :

Generators :

- Active and reactive powers
- Equivalent circuit parameters
- Inertia constants
- Reactive power limits

Transformers :

- Equivalent circuits parameters
- Tap values
- Bus number of the tap side

Transmission lines :

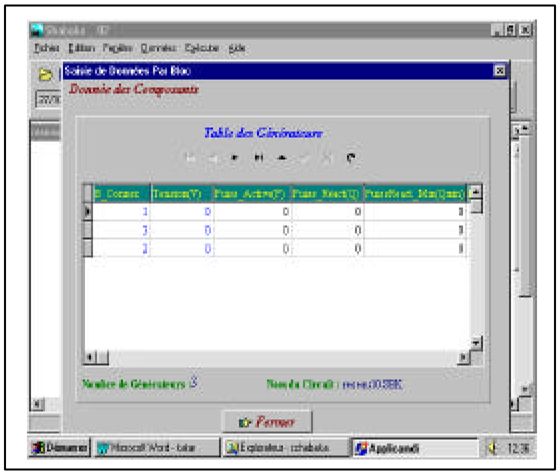
- Transmission line parameters
- Type

Motors :

- Equivalent circuit parameters
- Active and reactive powers
- Voltage

Loads :

- Active and reactive powers
- Voltage



Saisie de Données Par Bloc

Donnée des Composants

Table des Générateurs

B_Connex	Tension(V)	Puiss_Active(P)	Puiss_React(Q)	PuissReact_Max(Qmax)
1	0	0	0	1
3	0	0	0	1
2	0	0	0	1

Nombre de Générateurs 3   Nom du Circuit : reseau9.DBK

Fermer

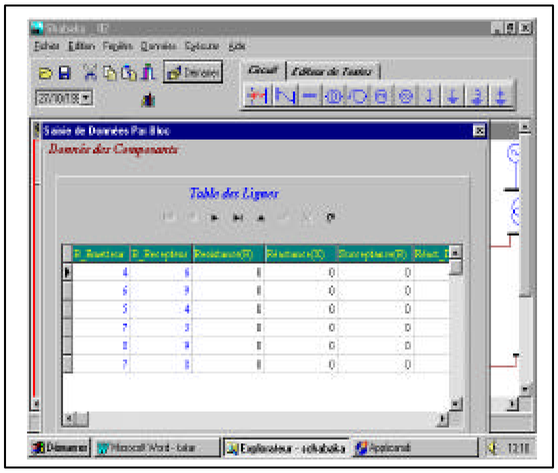
Figure 3 Data page of Generators

3.3External programs

The computing models incorporated provide the user with the possibility of running the power simulation functions such as Fault level and Transient stability or others.

The communication between GUI and the computing models is done through the DB. So for any change in the study network topology, the user can make any desired modification on the line diagram which will be transmitted to the DB and the computing model results.

Files of the simulation results are created and stored numerically and graphically and can be displayed on the screen either on the network configuration or displayed as curves.



Saisie de Données Par Bloc

Donnée des Composants

Table des Lignes

B_Emettrice	B_Receptrice	Resistance(R)	Réactance(X)	Susceptance(B)	Réact_I
4	6	[illegible]	0	0	
6	9	[illegible]	0	0	
5	4	[illegible]	0	0	
7	3	[illegible]	0	0	
8	9	[illegible]	0	0	
7	8	[illegible]	0	0	

Figure 4 Data page of Lines

4. System Implementation

The Object Oriented approach presented in this paper can be resumed as follows:

1. It is based upon physical power system objects.
2. Numerical algorithms are designed separately as application objects.
3. The application objects use the network objects as data stores by sending requests to the objects well defined interfaces. Due to encapsulation, internal details of network objects can be changed and new functionality can be added without affecting existent application objects.

4.1 Object oriented power system modeling

As described in the previous section, modeling of the power system is of central importance to the development of the system. In order to support a wide range of power system applications, power system should be modeled as basic or low level at various levels of abstractions.

By applying the object oriented technique, a hierarchical object model is built to describe the static structure of a power system. This object model was identified during the analysis and calibrated throughout the design stage. The class diagram of this model is shown in figure 5. As shown in this figure, the proposed object model is mainly based on the physical objects that exist in real power system, that is, electric devices. Each type of a device is represented as a class, which defines both the attributes and the procedures associated with this particular device.
The first step is to identify the relevant object classes of the problem domain and to specify their relations. So each power system component, which has to be considered, corresponds to a class in the object model. The class contains attributes that describe the state of an object and operations that define its behavior.

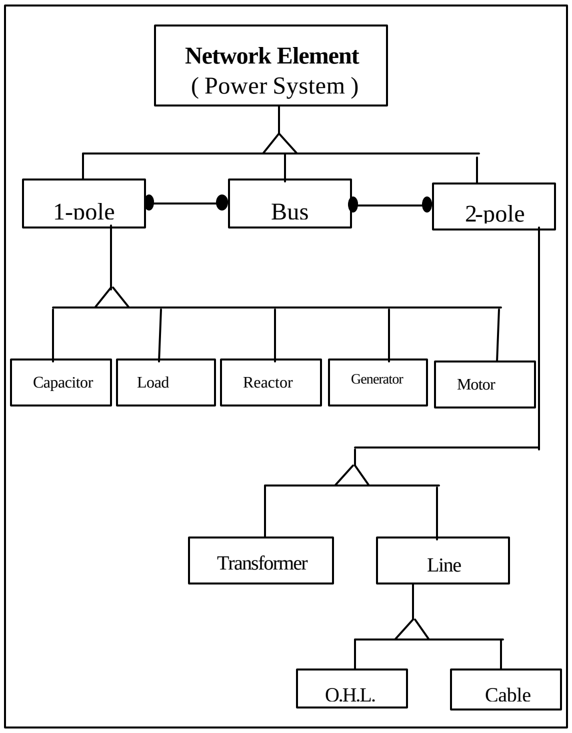


Figure 5 Object Model of the power system

4.2 Implementation of the designed power system object

Implementation is the final stage of the software development. At this stage, a particular programming language is used to implement the object model developed during the stage design. DELPHI is the object oriented programming language used for the implementation of the designed object models.

5. Conclusion

A power system package named SHABAKA providing aids to the power system dispatchers and electrical engineers has been presented in this paper. It combines text and graphics in one package to allowing a concise pictorial representation of aspects of power system analysis and design. It is implemented with object oriented approach using DELPHI programming language. The object oriented based development is improved by the easiness of flexibility, Expandability, maintenance and data integrity.
The results of its application to Load Flow, Fault level and Transient stability analysis demonstrate the interactive nature of its environment which can be extended in a future work to include new simulation functions and also to increase its capability to simulate large power system .

6. References

1. S. Li and S. M. Shahidehpour, ″An Object Oriented Power System Graphics Package for Personal Computer Environment″, IEEE Transactions on Power Systems, Vol. 8, N° 3, pp. 1054-1060, August 1993.
2. Kevin F. Chan and Jaques Ding, ″Interactive Network Planning and Analysis on a Personal Computer″, IEEE Computer Applications in Power, pp. 43-47, January 1990.
3. M. Belkacemi, L. Benfarhi, «A software Tool for Graphical and Numerical Identification of Power System Networks », Proceeding of the UPEC'98 Conference, pp. 739-742, 8- 10 Sept. 1998, Napier University, Edinburgh, UK.
4. Jun Zhu and David L. Lubkeman, ″ Object Oriented Development of Software Systems for Power System Simulations″, IEEE Transactions on Power Systems, Vol. 12, N° 2, pp. 1002-1007, May 1997.
5. Z. L. Gaing, C. N. Lu, B. S. Chang and C. L. Cheng,″ An Object Oriented Approach for Implementing Power System Restoration Package″, IEEE Transactions on Power Systems, Vol. 11, N° 1 pp. 483-489, February 1996.
6. Edmund Handschin and al., ″ Object Oriented Software Engineering for Transmission Planning in Open Access Schemes″, IEEE Transactions on Power Systems, Vol. 13, N° 1, pp. 94-100, February 1998.