Object-Oriented Design for Automatic Data Acquisition

Z. Mo, Department of Physics
Z. Chen, Department of Math and Computer Science
University of Nebraska at Omaha
Omaha, NE 68182

Abstract

We discuss software development for automatic data acquisition in a resistivity measurement system, and use it as an example to examine the object-oriented design principle for scientific computation. The literature in software methodology for scientific computing is relatively scanty so far, our paper will serve as a contribution to the exploration of this area.

1 Introduction

A tremendous spur on object oriented paradigm has come out recently. In this paper we examine the object-oriented design principle and related computational issues for scientific computation. Our discussion is on software development for automatic data acquisition in a resistivity measurement system.

Resistivity measurements are performed in various areas in either science research or industry. The study on resistivity behavior of various kinds of materials gives information about their microscopic structures, magnetic properties, phase transitions, etc. Generally, the results from resistivity measurements are shown as the relationship of resistivity vs. some external environmental parameters such as temperature, pressure, the direction and magnitude of external magnetic field, etc. However, the external parameters usually can not be well controlled. Therefore, automatic data acquisition controlled by a computer is necessary for resistivity measurements.

Z. Mo has developed the software which performs automatic data acquisition for a resistivity measurement system. It was used for resistivity measurements on the rare-earth compound system Ce(Pd_{1-x}Cu_x)Si_2 through the temperature range from 4K to 300K [4]. We are now in a process of examining the design methodology used in the software development, aiming at establishing the concrete guidelines toward "genuine" object-oriented design for automatic data acquisition, and generalize it for scientific problem solving.

The purpose of this paper is to report our ongoing investigation. Starting from a general discussion on the object oriented design and scientific computation, we examine how to apply this methodology to the software development of automatic data acquisition for resistivity measurement. Since literature in software methodology for scientific computing is relatively scanty, we hope this paper will contribute to the exploration of this area.

2 Object oriented design and scientific computation

The need to develop and maintain software systems in a competitive and dynamic environment has driven interest in new approaches to software design and development. The problems with the classical waterfall model is now generally recognized [6]. Problems with traditional development using the classical life cycle include no iteration, no emphasis on reuse, and no unifying model to integrate the phases [3].

A design paradigm is characterized by its view of the decomposition process in problem solving. The design philosophy of the object-oriented paradigm takes a modeling point of view. This allows the designer to work with one approach which begins in the problem domain and transits naturally into the solution domain. By building a model of the problem into the application system, the resulting design is more responsive to changes in knowledge about the problem situation. The modularity of these designs
and information hiding capabilities of most object-oriented languages contribute to the technique's responsiveness to modifications [3].

In the object-oriented design paradigm, the pieces of the design are objects which are grouped into classes for specification purposes. In addition to traditional dependencies between data elements, an inheritance relation between classes is used to express specialization and generalizations of the concepts represented by the classes [2].

To relate object-oriented paradigm to scientific computing, let us first examine some important characteristics of the latter. Scientific computing is a highly diverse activity spanning a multitude of disciplines. It draws on mathematics for answers, as well as a number of areas in computer science and computer engineering. As problems to be solved have become larger, the need to organize and manage the data of the problem has become more acute and data management techniques become increasingly important in scientific computing [5]. How to address and formulate scientific computation problem in terms of today's computer technology, particularly those related to software engineering, should become a focal concern of scientific computing.

Software for scientific computation shares lots of common considerations with other software, such as the need for modular design and the need for reuse. What is unique for scientific computation? Numerical computation should be incorporated into software development cycle. Speed and accuracy should be guaranteed. In a real-time system, scientific computation must be able to provide timely feedback for controlling purpose. To satisfy these requirements, scientific computation must be supported by good design.

The representation and management of data structures have been identified as a central concern in scientific computing. A good design for scientific problem solving thus implies handling data structures in an appropriate manner. In a paper published in the 70's, it is mentioned that most of the difficulties encountered in expressing scientific and engineering problems in computing arises from inadequacies in the data structuring, storage management and data management capabilities involved in computer programming [1]. The great achievements in the past decade have doubtlessly provided us better programming tools as well as high speed computing facilities, but a better software development methodology is still on demand. Since object-oriented discipline encourages organizing programs in terms of objects (i.e., data structures) rather than functions, it is particularly suitable for scientific programming.

3 Object oriented design discipline and automatic data acquisition

Automatic data acquisition for resistivity measurement requires significant amount of scientific computation. It is a good chance for us to investigate some important issues of object oriented design for scientific programming.

The components of the measurement system is configured in Fig. 1. The computer, running on Venix operating system, sends instructions to current source making it to input current to sample according to our needs, and instructs the nanovoltmeter to read voltage, so that the resistivity of the sample can be calculated. The R/G bridge has totally four channels, three of them are assigned to three resistance thermometers, respectively, while the other is assigned to a heater driver.

![Fig. 1](image-url)
The object oriented design is reflected in the following considerations.

1. In an object oriented paradigm, a class definition introduces a number of features representing operations applicable to instances of the class [3]. Particularly, we envision the software system consists of objects such as controllers, sensors, sample, and etc. Each of these objects is featured by some operations. For instance, the object controller is featured by the operations of giving instructions to heaters, and get feedback from the sample; and object therm (stands for thermometer) is featured by its operations of collecting data and reading the temperature changed due to heater.

2. The object oriented paradigm encourages the use of entity-relationship diagram (E-R diagram) to represent entities and their relationships [3]. Upon investigating this system, we have identified some important entities, such as "controller" and "sensor" (i.e., thermometers); as well as their relationships, such as "instruct" which connects entities "user," "sensor" and "meter". The E-R diagram is shown in Fig. 2, where entities are in [], and relationship names are in <>.

Practical reusable components should be organized around objects rather than functions. In fact, reuse of the software for automatic data acquisition can be realized at different levels. For example, the previously calculated result for resistance thermometer calibration interpolation had been stored in an external file, thus making the process of computation (handled by one module) and the use of the computation result (which is handled by a second module) separate. This makes the reuse of the second module possible.

Below we give some examples to illustrate the functions developed to realize our design considerations. To support the operations on the object "controller" a function is developed for reading in the electric current desired and sample size from an external file which can be modified by user for any sample. It also assign thermometer indices to be associated to R/G bridge channel. Another function is developed to read in constants for converting resistance of each of three thermometers to temperature and vice versa. The interpolation tried both polynomial fitting and cubic spline method and the optimal coefficients were selected for each thermometer. We also have function to turn the nanovoltmeter current source and R/G bridge on, initialize the appropriate parameters for these instruments and specify their functions, such as their working range, maximum range, automatic scan, etc.

The function "change-temp" is the actual process for automatic data acquisition. First, it sets working parameters for the heat driver including gain, filter, balance and enables it update. It then collects resistivity data at each specified temperature step. The power which driver supplies to heater is scanned regularly as the feedback and it is tested and then adjusted for optimum thus automatically controls the speed of temperature rising. Whenever a temperature data is read from a thermometer (the selection of thermometer is done automatically by switching bridge channel upon reading for different temperature region) current source sends a specified current to sample then nanovoltmeter reads sample's voltage, so that a pair of data for resistivity vs. temperature is acquired.

As an example of operation available on the object "terminal," the function "display" prepares a file "plot" for graphing the curve of sample resistivity vs. temperature so that after the experiment the user can see the resistivity behavior immediately.
4 Conclusion

The software development technology for scientific computation is relatively scanty. In this paper, through the software development of automatic data acquisition for a resistivity measurement system, we have explored some considerations of object-oriented design for scientific computation. Our discussion indicates that there is a great potential for object-oriented approach in scientific computing. In addition, it is hopeful that object-oriented design methodology will make it possible to incorporate knowledge-intensive techniques as developed in artificial intelligence (AI).

The software developed for automatic data acquisition, as reported in this paper, is not a product of fully object-oriented programming by itself. The software was written in C, which is not an object-oriented language. However, the software can be re-implemented in C++, an object-oriented language.

References


