On the Design and Implementation of User-friendly Interface for Scientific and Engineering Applications

W. SUN, Y. CHEN, H. BOUSSALIS, C. W. LIU, K. RAD, J. DONG
Electrical and Computer Engineering
California State University, Los Angeles, CA 90032, USA

Abstract: - Sophisticated visual programming environments facilitate the implementation of graphic user interface (GUI) from the programming aspect. However, how to design an effective and efficient GUI to meet the requirements of large-scale scientific and engineering software still remains an interesting research topic. This paper presents a methodology of GUI design for engineering applications. First, the requirements and the tasks of the specific engineering application should be carefully analyzed. Second, a prototype of the GUI is generated. Then, the concept of interactive GUI design is employed to improve the user friendliness. Finally, extensive testing is conducted, and iterative software design method is applied to improve the GUI design based on the testing results. The presented methodology is employed in the design procedure of the user-friendly graphic user interface in a 3-D animation system for a large space segmented telescope test-bed.

Keywords : - Graphic User Interface, interactivity, user-friendliness

1 Introduction

Recent developments in software architecture present new opportunities and challenges for user interface design. On one hand, sophisticated visual programming environments, such as Microsoft Visual Studio, etc., facilitate the implementation of graphic user interface (GUI) from the programming aspect. On the other hand, how to design a good GUI still remains an interesting research topic. Many research efforts have been made to address this question [1,2,3]. In general, GUI design consists of the following phases: user requirement analysis, task analysis and modeling, GUI prototyping, and usability evaluation. However, to design an effective GUI for large-scale scientific and engineering applications, the above deign methodology is not sufficient. To model the tasks of a sophisticated scientific or engineering system, a large number of parameters are necessary.

Therefore, users without background knowledge may get lost when confronting the inputs/outputs. Or, they may get confused about the steps needed to go through before getting a result. In our paper, the methodology of GUI design is improved to achieve better user friendliness. Taking the specific concern of scientific/engineering applications into accounts, our proposed GUI design procedure is tuned to reduce the user errors and increase the efficiency of the software.

To illustrate our proposed methodology, the design procedure of the user-friendly GUI for a 3-D animation software package of a large space segmented telescope test-bed is presented as an example. As a NSF-sponsored project, the goal of developing the 3-D animation system is to demonstrate the effect of various control algorithms proposed in our previous research [5,6], and to disseminate the knowledge to the public. Hence, a well-designed user-friendly interface is essential to fit the needs of audience with different professional background.

The rest of the paper is organized as follows: Section 2 provides an overview of the GUI design methodology. The details of the design of GUI for two engineering applications are presented in section 3. Section 4 described the implementation steps as well as the results. Section 5 concludes the paper and shows the future work.

2 GUI Design Methodologies for Engineering Software

The design procedure of a generic user-friendly interface, from the aspect of software engineering, takes four major steps: 1) user requirement analysis, 2) user objects modeling, 3) GUI prototyping, and 4) usability evaluation [4]. Through the first two steps, the requirements of usability is examined, an abstraction of user requirements is created, and user objects model of the system is established based on the user action and the corresponding system responses. Step 3 and 4 creates and evaluates the GUI for the system. The design procedure works in an iterative fashion.

To meet the requirements of engineering application software, additional steps should be added to the generic GUI development cycle. Figure 1 depicts the improved GUI design procedure for engineering applications.

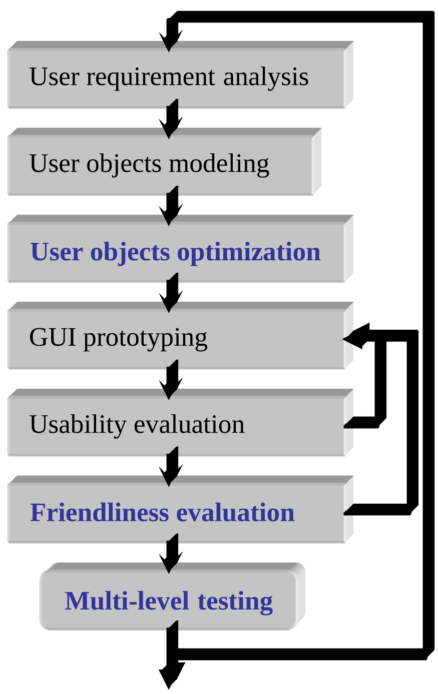


Fig. 1- GUI design procedure for engineering software

To better fits the needs of engineering applications, one optimization step is added to refine the user objects model generated by user requirement analysis. Since the GUI interacts with multiple complicated modules to support the engineering core, a clear and well modularized interface is essential. Another additional step is friendliness evaluation. In engineering applications, to complete a given task might be very difficult to the users. Therefore, it is more important to provide an easy-to-use GUI for engineering software. The feedback of the friendliness evaluation will be used to improve GUI prototyping. This iterative process will continue until the evaluation results are satisfactory. In addition to the added steps in the design cycle, the testing process is also revised. The GUI will be tested by multiple user groups with different levels of knowledge related to the specific engineering application, and the feedbacks help improve the GUI design to meet the requirements of users with different background

3 GUI Design for 3-D Animation of Large-scale Segmented Space Telescope

In this section, the design procedure of a highly user-friendly GUI for a 3-D animation system of large-scale segmented space telescope is presented. Emphasize of the presentation will be focused on the how to increase the modularity and user-friendliness in GUI design for engineering applications.

3.1 Background

For future space-borne astronomical missions, a segmented space reflector telescope is preferred rather than a monolithic one. To mimic the optical properties of a monolithic telescope using a segmented one, an effective real-time control system has to be established for shape control and precision pointing. Funded by NASA. A segmented reflector test-bed has been built at the Structures Pointing And Control Engineering (SPACE) Laboratory at California State University, Los Angeles, based on which, several efficient algorithms have been developed to address the problems associated with the real-time control of a large segmented optical system.

In order to visualize the effect of the real-time control algorithm, a 3-D animation system for the segmented space telescope was developed. Nevertheless, a well-designed GUI is essential to allow the interaction between the users and the control algorithms. Figure 2 shows the controller implementation of the physical space telescope testbed. The digital controller for panel shape control employs one of several control algorithms such as PID control, H-infinity control, Adaptive, and neural network control, etc. Figure 3 is a conceptual block diagram of the closed-loop system that represents the controller implementation of the physical system in Figure 2.

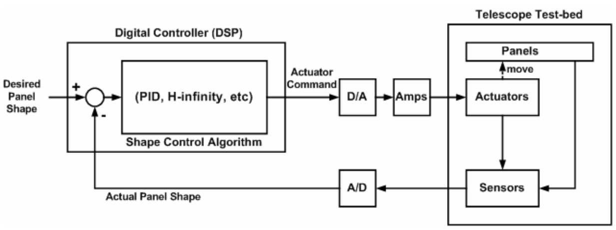


Fig. 2 - Real-time control implementation

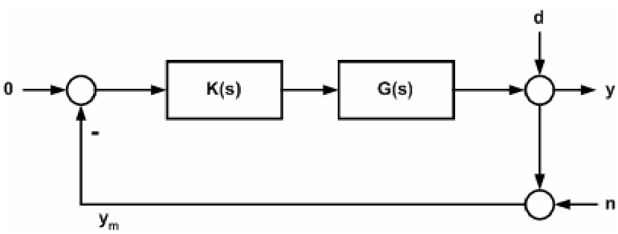


Fig. 3 - Block diagram of the closed-loop control system

In the block diagram, G(s) is the transfer function of a physical system. K(s) is the implemented control algorithm. y, ym, d and n are the actual virtual displacements, measured virtual displacements, disturbance, and measurement noise respectively. Based on [1], the state space realization of a peripheral segment can be represented as

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} A_1 & 0 & 0 \\ 0 & A_2 & 0 \\ 0 & 0 & A_3 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} B_1 & 0 & 0 \\ 0 & B_2 & 0 \\ 0 & 0 & B_3 \end{bmatrix} \cdot u$$

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} c_1^T & 0 & 0 \\ 0 & c_2^T & 0 \\ 0 & 0 & c_3^T \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} d_1^T \\ d_2^T \\ d_3^T \end{bmatrix} \cdot u$$

The above equations were derived from the transfer function G(s) with the experimental measurements of the matrices of coefficients based on the test-bed movement.

In order to obtain a good user objects model, an in-depth analysis of the control algorithms is necessary. First, internal (transparent to the users) and external (inputs and outputs) parameters of the system should be distinguished. Second, decision should be made through user requirement analysis that which external parameters allow user interaction. For example, in the control system represented by above equation, u, y, d (input, output, and disturbance) are external parameters. Since one of the user requirements is to view the effect of control algorithm under various disturbances, d should be adjustable by the users.

3.2 Modularity in Engineering Software Development

As described in section 2, the user objects model should be further optimized. Due to the complexity of engineering software and the huge cost associated with maintenance and re-development, modularity is one of the most importance criteria for optimization.

Figure 4 shows the modularized design of the 3-D animation system for space telescope [7].

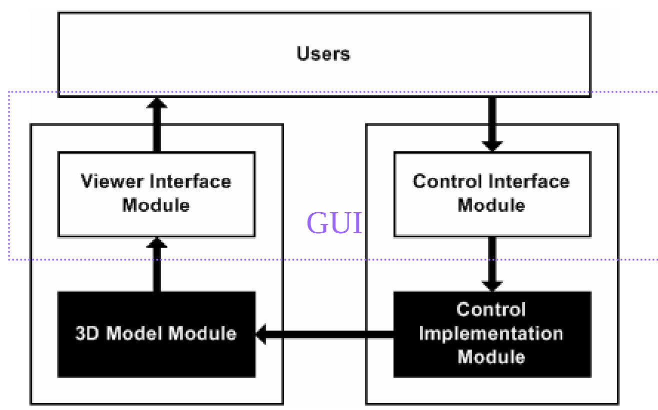


Fig. 4 - Modulization of the animation system

The four modules are described briefly as follows:

3.2.1 The Control Interface Module

The control interface module allows users to setup control algorithms and application-specific input disturbances. The control interface module defines the formats of inputs, collects necessary data and parameters from the user, and then sends them to the control algorithm module. The purpose of this module is to support a well-defined and user-friendly interface. The other components shielded by the interface can be kept intact to further ease the design of the system.

3.2.2 The Control Implementation Module

Transparent to the users, the control implementation module is does the major computation works. This module receives the data or parameters from the control interface module and computes results using a control algorithm chosen by the user. The control implementation module simulates all the control devices of the telescope test-bed and generates decentralized control in an iterative fashion.

3.2.3 The 3-D-Model Module

The OpenGL routines in the 3-D-model module receive vertices and compose 3-D models using basic geometric primitives such as points, lines and polygons. The kinematics properties of 3-D model also define in this module using OpenGL modeling transformations. For keeping this module independent from mathematical computing, we create geometric primitives and define kinematics properties only and leave input ports of transformations for computing results of control implementation module.

3.2.4 The Viewer Module

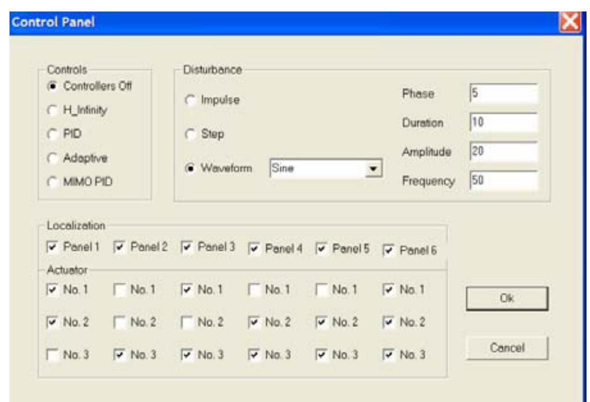
The viewer module is a user interface for showing the animation of 3-D telescope models. For observing the animation, the viewer module should have basic viewing functions such as zoom, rotation and pan.

In our design, the GUI is split into two independent modules: the control interface module and the viewer module. This design maximizes the modularity of the software, and simplifies the GUI which makes it easier to use.

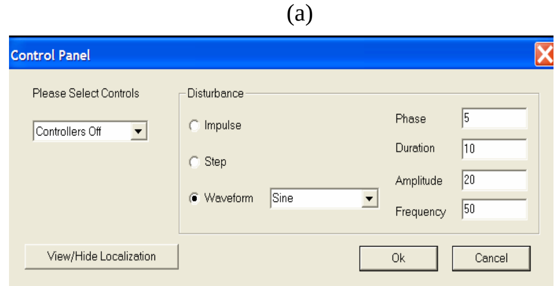
3.3 Increase the User-friendliness in Engineering Software

User friendliness is a serious concern in engineering software design. A user may be confused when presented with a large number of inputs and outputs. If the GUI is designed without considering the user friendliness, a user may not be able to use the software without proper training. Thus, in our proposed GUI design cycle, user friendliness evaluation is applied iteratively to increase the viability of the software.

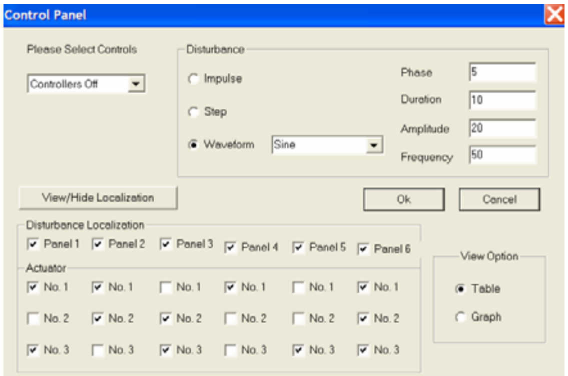
Here an example is used to describe the iterative process of GUI prototyping and user friendliness evaluation. Figure 5 (a) shows the first prototype of the control panel. There were several problems associates with this design. First, it allows users to select control algorithms using a group of radio buttons. Apparently, this “short-sighted” design did not consider that more control algorithms might be added in the control implementation module. A solution is to replace the radio buttons with a drop-down box that can accommodate new items easily. Second, the GUI shown in Figure 5-(a) uses a static chart table of check boxes to allow users to select the locations of disturbances. After the first round user friendliness evaluation, the feeling is that this design is too complicated to use and may cause confusion to the users. To improve the GUI prototype, a set of default value is set for the users such that the user will not go through the complicated selection procedure for each simulation. In addition, the concept of dynamical design is incorporated here to simplify the GUI. As shown in Figure 5-(b), a show/hide button for the localization of the disturbance to the panel and actuators is added for dynamical display of the table of check boxes. In default setting, the complicated table of check boxes is not shown (as Figure 5(b)). Whenever the user want to do the selection by themselves, they can click on the show button and the chart table of check boxes will appear on the extended bottom part in the control panel, as shown in Figure 5(c). After selection, the user can hide the table by pressing the view/hide button again.



(a)



(b)



(c)

Fig. 5 - Illustration of GUI improvement by user-friendliness evaluation

Although the revised GUI prototype is less complicated after the first round user friendliness evaluation, it is still not so easy to use for the users without good knowledge of space telescope. Hence, after the feedback of the second round user friendliness evaluation, the prototype of control panel is further modified. As shown in Figure 6, instead of using a table of check box for location selection, the new design allows users to directly click on the desired location in the graph of telescope panels. Apparently, this graph-based selection is more user-friendly.

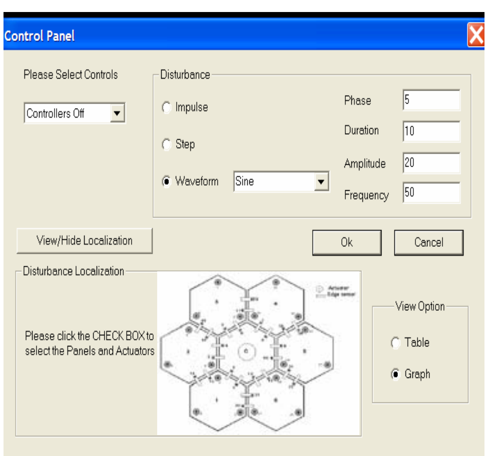


Fig. 6 - Revised control panel prototype after 2nd round user friendliness evaluation

4 Implementation and Results

The GUI of the 3-D animation system of large-scale space telescope is implemented using Microsoft Foundation Classes (MFC). Due to the modularized design, the GUI for interacting with control algorithms and the GUI for viewing 3-D animation are implemented separately. The window snapshots for control panel (GUI for interacting with control algorithms) are similar to the prototype in Figure 6; the window snapshots of GUI for viewing animation are displayed in Figure 7, in which a 3-D model of a peripheral segment is shown on the viewing window with three slider bars that control three actuators. When slider bars are moved, the actuators drive the panel to new position.

5 Conclusion

This paper presents the methodology of GUI design for complicated engineering applications, with a focus to optimize software modularity and user friendliness. As an example, the design procedure of a highly user friendly GUI of a 3-D animation software for large-scale space segmented telescope is used to demonstrate the effectiveness of our proposed methodology. The GUI is implemented using Visual C++ in windows platform. It is integrated with OpenGL to realize the interface for viewing 3-D animations.

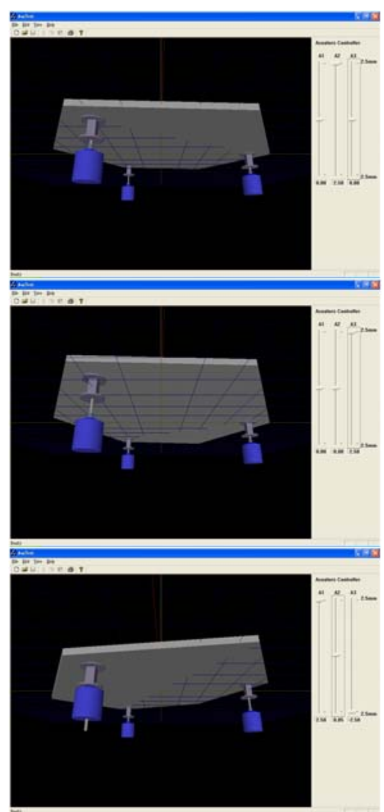


Fig. 7 - GUI for viewing animation: slider bars control the movement of the panel

6 Acknowledgments

This work was supported by NSF under Grant # EEC0121026.

References:

[1] C. Phillips, Towards a task-based methodology for designing GUIs, Int. Conf. On Software Engineering: Education and Practice, Jan. 1996

[2] T. G. Sparkman, Lessons learned applying software engineering principles to visual programming language application development**,** The 23rd Annual International Computer Software and Applications Conference, Oct. 1999

[3] L., Charest, at el, A methodology for interfacing open source SystemC with a third party software, conf. On Design, Automation and Test in Europe, March 2001

[4] D. Redmond-Pyle, Software development methods and tools: some trends and issues**,** Software Engineering Journal, Vol. 11, March 1996

[5] H. Boussalis, M. Mirmirani, Z. Wei, Decentralization and PID Controller Design for Large-Spaceborn Telescopes, IASTED International Conference on Application Modeling, Simulation and Optimization, Pittsburgh, 1995.

[6] H. Boussalis, M. Raghavender, The Application of H-infinity Control Law to a Decentralized Segmented Reflector, *IASTED/SCS Conf.*, Australia, 1996.

[7] Y. Chen, C. Liu, H. Boussalis, et al., *3-D Animation for a Segmented Space Reflector Telescope,* the 7th World Multiconference on Systemics, Cybernetics and Informatics (SCI 2003), Orlando, USA, July 27-30, 2003.