Specification and Automatic Generation of Intelligent Graphical Interfaces

Yukari Shirotal
and Toshiyasu L. Kunii

[1] The Laboratory of Theoretical Physics, Faculty of Science, Gakushuin University
1-5-1 Mejiro, Toshima-ku, Tokyo 171 (JAPAN)

[2] Department of Information Science, Faculty of Science, The University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo 113 (JAPAN)

Abstract
This paper outlines a new type of visual interface called Enhanced Menu-Based Software (EMBS), and describes a visual specification method for such software. In EMBS, data management facilities of form-oriented systems, automatic recalculating facilities of spreadsheets, and CAD facilities are successfully integrated. The Program-Specification-by-Examples paradigm and visual programming by icons are key factors that facilitate the development of the above software.

1. Introduction
The importance of visual interfaces has been widely acknowledged, and has motivated the development of various systems and techniques. This paper describes a visual interface named Enhanced Menu-Based Software (EMBS) that we have developed, and a generation system for such software. The EMBS serves as a visual interface for application procedures. Facilities of form-oriented systems to handle underlying databases, spreadsheets, and CAD systems are successfully integrated by the constraint-oriented paradigm. A description of the database facilities and an action propagation model for the EMBS can be found in [10,11]. In this paper, we focus on the EMBS intelligent CAD facilities.

Generally, in the design process of a certain application, there are various constraints which define the design objects, and these constraints should be maintained during an interactive graphical dialogue. In order to reduce the work load of a user, the integration of the constraint-oriented paradigm with visual interactive techniques seems beneficial. A variety of intelligent CAD systems equipped with the constraint-oriented paradigm have been developed. Notable among them are SketchPad [14], Thinglab [1,2,3], Juno [7], Animus [4], Alternate Reality Kit [12], and GRAFLOG [8,9]. In [13], a graphical simulator which supports knowledge-based instruction of Navy steam engineers is presented.

In the EMBS, the users are allowed to specify the geometrical constraints and relations in action definitions. As a result, if any cell or graphical element is changed, the related cells and graphical elements will be automatically reevaluated to satisfy these relations.

An overview of the EMBS and its generation process will be given in the next section. In Section 3, we explain how the users define constraints in the EMBS specification language. To illustrate these methods, we use an EMBS for determining air conditioning load as an example. Finally, Section 4 summarizes this paper.

Fig. 1. The Enhanced Menu-Based Software (EMBS) structure and the action propagation example. An EMBS consists of menu-screens and the links between them. The invoked actions retrieve data from cells on other menu-screens and from underlying database and perform calculations.
2. Overview of the Automatic EMBS Generation System

An EMBS consists of menu-screens and the links between them (see Fig. 1). A menu-screen is an image displayed on a terminal screen, and consists of the following three image elements: (1) cell image: an area enclosed by "[" and "]", (2) button image: a boxed area, and (3) background image: all images except cells and buttons. The EMBS action invocation is triggered by clicking a button or by changing the value of a cell. Following the initial action invocation, subsequent actions are executed in turn and the dependent cells are automatically reevaluated. Moreover, the EMBS incorporates data manipulation facilities. The users can perform database operations through cells, which can be treated in the same manner as other cells from the users' point of view.

The EMBS generator permits users to specify their own EMBS without the aid of software specialists. The users create EMBS specifications to define their requirements. The specifications consist of two definition parts: a screen definition part and an action definition part. The EMBS generator then processes the specification to produce program code that performs the desired function. The generator consults the action function library and extracts action functions necessary to compile the specification into a working program. The action library contains predefined action functions written in a conventional programming language such as C, and is made available to the users.

In the screen definition part, the example value placed in a cell or a button (e.g., "Mr. Brown" or "$150.00") is registered as the cell or button identifier used as such in the following action definition part. This example-oriented approach offers greater understandability and helps the users shape their ideas into working programs. We call this specification method Program-Specification-by-Examples.

An action in the action definition part is specified by a triplet, called an action statement:

<trigger_list> <action_function> <object_list>. In the EMBS specification language, there are three types of action statements as shown in Fig. 2: (1) action statements with objects; (2) action statements with an empty object list; (3) arithmetic expressions. The users can define an action function macro as a sequence of the action statements. There are only a limited number of reserved words, and all reserved words are represented as strings of symbols, rather than alphabetical characters (e.g., "<" and ">"). Therefore, contrary to conventional programming languages, it is easy to master the EMBS specification language. In addition, because the example values serve as cell or button identifiers, the users can use their mother tongue for the identifiers; i.e., the EMBS specification is language-independent.

**BNF grammar for the action**

< action statement >
  --> < trigger list > "--<" < action > "-->
  --> < trigger list > "--<" < action > "-->
  --> < expr > "--<" < object >
  < trigger > --> < button >
  < object > --> < cell >

**Examples:**

DINERS AMEX MC <-- select() --> VISA
8208-0166 VISA <-- find(card_member) --> John-Brown 02/90
($150.00 + $30.00) / 3 --> $60.00

![Fig. 2. The BNF grammar for the action statement with examples.](image-url)
3. An EMBS for Determining Air Conditioning Loads

In order to incorporate intelligent CAD facilities, we have extended the EMBS and its specification methods, so that the users can also define actions that manipulate the CAD displays in relation to data by using the iconic programming paradigm. As a case study, this section describes the specification of an EMBS for determining air conditioning loads. In air conditioning planning, an equipment designer uses data from a variety of databases. Fig. 3 and 4 present some menu-screens of the EMBS through which the designer accesses the database and calculates the cooling loads.

A psychrometric chart is an essential tool for the designers to estimate the air conditioning parameters. Points on the chart are represented as a point on the chart or by the following parameters:

- (a) dry bulb temperature: \( t[^\circ C] \);
- (b) wet bulb temperature: \( t[^\circ C] \);
- (c) relative humidity: \( \phi [%] \);
- (d) absolute humidity: \( x [kg/kg] \);
- (e) enthalpy: \( h [kcal/kg] \).

If at least two of the above parameters are given, the air state can be determined and the other parameters can be calculated. Fig. 5 presents an example menu-screen of the EMBS equipped with CAD facilities for the psychrometric chart. It shows a template pattern for designing a steam or spray heating facility. There are five factors to be considered:

- \(<1>\) an outer air point,
- \(<2>\) an inner air point,
- \(<3>\) a mixture point of the outer and the inner air,
- \(<4>\) a steam air point for heating and humidification, and
- \(<5>\) an air point at the air outlet.

To define the target EMBS, the users create the EMBS specifications to define their requirements through windows (Fig. 6). The large window on the left corresponds to the menu-screen of Fig. 5. In the screen definition, the users draw the psychrometric chart image composed of scales, axes, lines, points, and so on by using painting tools.

The static parts such as axes are registered as the background image. The dynamic parts are registered as icons in an icon box. The users can define the icon image by grouping a set of primitive graphical elements. \( \mathbb{1}, \mathbb{2}, ..., \mathbb{5} \) in Fig. 6 are user-defined icons. It is also possible to construct more complex icons, such as an image of a steam pump or an air outlet. Then, the users give aliases for the registered icons, and specify the action statements by using the aliases. In the smaller window shown in Fig. 6, aliases such as \(<1>\) and \(<2>\) are used to identify the corresponding air state points.

Now, let us consider the geometrical constraints that apply in Fig. 5. In the following explanation we shall first present the contents of a constraint, and then, the corresponding action function macro will be given as an action statement sequence, where we identify the air state points by \(<1>\), \(<2>\), \(<3>\), \(<4>\), and \(<5>\).

### <constraint 1>
Points \(<1>\), \(<2>\), \(<3>\), \(<4>\), and \(<5>\) on the chart are represented by the four parameters: i.e., \( t \) (dry bulb temperature), \( \phi \) (relative humidity), \( x \) (absolute humidity), and \( h \) (enthalpy).

#### <action function macro 1>

\[
<1> -- > mouse \_xy () > > > 0.0 0.00150
<2> -- > mouse \_xy () > > > 20.0 0.00726
<3> -- > mouse \_xy () > > > 14.8 0.00575
<4> -- > mouse \_xy () > > > 15.6 0.00779
<5> -- > mouse \_xy () > > > 27.2 0.00778
0.0 40.0 0.00150 0.90 -- > moist \_air () > > > <1>
20.0 50.0 0.00726 9.20 -- > moist \_air () > > > <2>
14.8 55.5 0.00575 7.04 -- > moist \_air () > > > <3>
15.6 70.0 0.00779 8.43 -- > moist \_air () > > > <4>
27.2 34.0 0.00778 11.27 -- > moist \_air () > > > <5>
\]

### <constraint 2>
The mixed air \(<3>\) is composed of the outer air \(<1>\) and the inner air \(<2>\) in the ratio 0.262.

#### <action function macro 2>

\[
0.0 \times 0.262 + (1 - 0.262) \times 20.0 --> 14.8
0.00150 \times 0.262 + (1 - 0.262) \times 0.00726 --> 0.00575
0.90 \times 0.262 + (1 - 0.262) \times 9.20 --> 7.04
(20.0 - 14.8) / (20.0 - 0.0) --> 0.262
\]

From this constraint, it can be seen that \(<3>\) always lies on the line \(<1>\) \& \(<2>\). If the ratio is not changeable, \(<3>\) will be changed automatically, whenever \(<1>\) and \(<2>\) are moved. The above "1" is a constant value; "1" did not appear in the cells of the screen definition of this menu-screen (see Fig. 6).

### <constraint 3>
The incline of the line \(<2>\) \& \(<3>\) is \(0.833\), and the temperature of \(<5>\) is \(27.2[^\circ C]\).

#### <action function macro 3>

\[
20.0 + 25000 / (0.24 \times 14500) --> 27.2
9.20 + 25000 / (14500 \times 0.833) --> 11.27
25000 / ((11.27 - 9.20) / 14500) --> 0.833
\]
Fig. 3. A menu-screen of an EMBS for the calculation of cooling loads. On the screen, immediately after the user inputs the direction, types and areas of the room elements such as walls and windowpanes, the EMBS retrieves a unit value of the heat load and the related coefficient K (e.g., sunlight shielding coefficient) from the database, and displays them. And then, the heating load is automatically calculated from these figures.

Fig. 4. A menu-screen of an EMBS to insert new meteorological data. In Japan, the Society of Heating, Air Conditioning and Sanitary Engineering of Japan provides a large collection of meteorological data for several main cities[6].

Fig. 5. A menu-screen of the EMBS equipped with CAD facilities for the psychrometric chart.
The incline of the line is 641, which represents the enthalpy of the steam, and always is on the same level as action function macro 4:

\[
14.8 \quad 0.00575 \quad 641 \quad --\text{spray}(O)--> \quad 8.43
\]

\[
14.8 \quad 0.00575 \quad 8.43 \quad --\text{incline}(O)--> \quad 641
\]

\[
0.00778 \quad --> \quad 0.00779
\]

The last action statement expresses an assignment.

The EMBS specification grammar prohibits cyclic definitions of action statements such as "A --< \text{funct1}(O)--> B" and "B --< \text{funct2}(O)--> A" to avoid endless loops. However, in order to allow bidirectional constraints, we permit the specification of cyclic action propagation, provided the cyclic action statements are in the same macro. The termination rule reads "if the re-evaluated value for a certain object element is (nearly) equal to the current value, the action propagation process is terminated."

In general, there may be several possible calculation routes in the above-mentioned action function moist_air. For example, if \( r \), \( \phi \), and \( x \) are given to calculate \( h \), the following three routes are possible: (1) \( r + h \), (2) \( \phi + h \), and (3) \( x + h \). In such a case, the function will automatically select the best route that leads to the minimum computational error.

During an EMBS session, the cells whose example values appear in the constraints will be recalculated according to the movement of the graphical elements, and conversely, cell value changes may be cascaded to the graphical elements. To declare a cell value to be floating (movable), the users may input a "?" to the cell. Graphical elements can also be defined as floating; the floating elements will turn red. This declaration is not permanent, and the users may change the mode between "floating" and "fixed" as they wish.

4. Conclusions

This paper described the Enhanced Menu-Based Software (EMBS) and the associated specification methods, focusing on its intelligent CAD facilities. For example, in the air conditioning loads calculation, the equipment designers require a visual interface with the following capabilities: (1) quick information retrieval; (2) convenient calculation tools; and (3) CAD facilities to display the parameters graphically. In the EMBS, these capabilities, i.e., 2-D CAD facilities, data manipulation facilities, and automatic recalculation capability of spreadsheets, are successfully integrated. In addition, because the EMBS incorporates the constraint-oriented paradigm, the users can define and share the constraints expressed as an action function or as an action function macro.

Prospective application areas span all situations involving users not familiar with computers. The EMBS can easily be tailored to suit the users' requirements, and the users can specify their own EMBS without the aid of software specialists.

Fig. 6. During specification of the target EMBS, the user creates the specification to define its requirements through windows.
References


