FLAIR - USER INTERFACE DIALOG DESIGN TOOL ✓

Peter C. S. Wong
Eric R. Reid

TRW Defense and Space Systems Group
Redondo Beach, CA 90278

ABSTRACT

To aid the system designer in achieving early involvement of the user, FLAIR (a user interface dialog design tool) was conceptualized and developed. FLAIR allows the designer to rapidly prototype a system's Man-Machine Interface.

A system designer can select a desired mix of input/output devices ranging from voice to high resolution graphics equipment. FLAIR prompts its users with a dynamic menu according to a predefined English-like syntax. Commands are entered by the designer's own voice. Entered commands are validated by voice pattern recognition and command language-gating. System messages and acknowledgements are presented via the monitor screen and/or a synthesized voice. Pointing devices are used to locate, place and/or pick objects from the RAMTEK 9400 25" high-resolution color monitor. The entire setup is hosted on a VAX 11/780.

The graphics display portion of FLAIR is handled by a Core Standard graphics package. A Core package, by definition, fits the needs of this tool very well with its definition of retained segments, windowing and viewports, and graphics device independence. This particular package also fits the specific needs of FLAIR in that it makes available most of the RAMTEK 9400's hardware features. These features include hardware pan and zoom, as well as basic image processing capabilities.

A relational DBMS has been integrated into FLAIR in order to manage system and user-defined data relationships. The user-defined data can be associated with a particular on-screen graphics symbol, and can then be queried at a later time.

In addition to single graphics snapshots (or "static frames"), FLAIR allows the system designer to create command menu hierarchies (or "dynamic scenarios"). This allows the designer to simulate, through client menu item selection, the system control flow. He can, in effect, create a tree of menus, traverse the tree at will, and select more menus or system actions as desired. All menus appear as sensitized areas on the graphics screen, and can be selected via any of the available input devices.

It is estimated that a system designer can greatly improve his dialog design productivity by using FLAIR. The designers need not code any formal programs. The designer need not master the usage of the host computer, the graphic input/output systems, the menu controls, or the data base programming before his designs are realized. FLAIR is there to assimilate, assemble, save and exercise his instructed operator dialog.

KEYWORDS: Man-Machine Interface, Computer-Aided Design, Dialog Design Language, Voice Synthesis and Recognition, MMI Testbed, Core Standard Graphics, Relational Data Base, Spatial Data Management

The personnel who will be the ultimate users of a system are the most important clients for designers of that system. An operator dialog interface that can promote the productivity of the user clients should be one of the goals of the system design. This goal can be accomplished by the clients' early involvement in the dialog planning, design, simulation and development (11). Dialog streamlining via simulations and final agreement with the client to the functional dialog should be achieved before production codes are generated.

To meet this objective, a Man-Machine Interface (MMI) testbed has been built at TRW. The testbed is fully operational. Included in the system are some of the latest technologies in computer graphics (software and hardware), voice recognition and synthesis units, a relational data base management system, picture capture equipment and interactive input devices. Elements of the MMI hardware are grouped into "workstations." Each station consists of a full complement of input/output peripherals that may be associated with an operator station. This is illustrated by Figure 1. The system designer, using one of these stations, can "mix and match" any of the devices to form the project's conceptualized console. The workstations themselves are connected to a VAX 11/780 computer. The MMI workstation control is provided by a software system generically referred to as Dialog Design Language (DDL). The DDL is the software cornerstone of the testbed. It enables the users of the system to design operator dialog in a simple, straightforward manner.

Following is a description of the DDL implementation in the testbed.

MOTIVATIONS AND CONCEPTS FOR THE DDL

The Dialog Design Language (DDL) is a comprehensive software tool that enables the system designers to express operator/computer interactions and the human performance researchers to conduct MMI research. The DDL is constructed so that a user can take advantage of all the available hardware installed in the Man-Machine Interface (MMI) testbed; yet, users do not have to master the intricacies of the host system before becoming a proficient user of the testbed. In this way, the designer can concentrate on the dialog designs and demonstrate various MMI subsystem concepts before a proven MMI subsystem is actually implemented.

By providing the end users with a chance to interact with a portion of the 'deliverable system' through scenario simulation, the DDL can lead to more accurate definition of system requirements. Various formats, content and sequence of interactions, as well as combinations of interactive devices to be used, can easily be considered. The usefulness of the DDL does not end with the establishment of system requirements, but continues throughout the design and development process.

For the human performance researchers, the DDL offers an easy method to construct experiments. Given the simple manner in which a chart or sequence of charts can be constructed and replayed under computer control, experiments can be constructed or modified without the cost and delay of using professional artists to illustrate the displays. Since the DDL can logically control the sequence of the displays, man-machine and human performance experiments will be more realistic and the data gathering activity more uniform from subject to subject.

CONCEPT: USER REQUIREMENTS

To establish the requirements for the Dialog Design Language, system designers from a number of TRW projects were consulted. In addition, the design has been reviewed with potential users, who plan to incorporate the DDL into their project work.

No matter how sophisticated and complete a MMI testbed may be, it is only as powerful as the system designer's ability to use it.

Since the DDL is the system designer's tool for invoking the powers of the testbed, it should be a model of utility. Out of this emerge two major requirements: 1) The DDL must be easy for the system designer to use. If systems designers are to be encouraged to use new tools, they must be convinced that their jobs will be simplified. 2) The DDL should be flexible enough to simulate a broad range of scenarios. Much effort and expense is involved in building a good MMI development facility. It would be a waste of resources if the DDL accommodated only a very limited set of applications. The philosophy behind the DDL, then, is to keep it user-oriented, natural in its structure, yet powerful enough to be useful. Most significant, the DDL should itself be a good example of MMI. The DDL is not strictly a language, but a menu-driven system that aids and directs the designer through a coherent and orderly translation of his scenario into a form that is executable in the MMI testbed.

The following major requirements form the basis for the initial structure of the Dialog Design Language (10).

A. Display Static Frames - be able to:

1. Construct a picture;
2. Edit a picture;
3. Save and retrieve a picture.

A picture can be in color. It can consist of background maps or other non-erasable data, symbols, geometric elements (points, lines, circles, etc.), and textual information.

B. Frames Scenario Dialog - be able to:

1. Display a sequence of frames;
2. Control the sequence of frames logically as a result of operator inputs;
3. Control the sequence of frames via predefined control hierarchy of menu response or time;
4. Save, edit, retrieve the control hierarchy.

The input devices shall be interactive. These devices shall be able to interchange with each other so that the system design can choose the devices he wishes to include in the final system.

C. Dynamic Frame with Scenario - be able to:

1. Modify frames as a function of operator/computer interaction;
2. Attach relations with individual symbols;
3. Include user algorithms to control the scenario.

D. Multi-Workstation Coordination-be able to:

1. Effectively utilize the MMI testbed resources from multiple workstations;
2. Simulate multiple workstation communications with each station interacting with certain portions of a scenario.

E. Code Transportability and Standards:

Since it is envisioned that DDL will be hosted and used at other installations, the tools should be able to:

1. Move the host independent modules more easily from one machine to another;
2. Have configuration management for updates.

THE DDL USER'S INTERFACE REQUIREMENT

The initial frame construction, subsequent editing and control is designed so that a system designer or human performance researcher can fully utilize the testbed without a major learning effort. The Dialog Design Language is menu-driven so that it can prompt a user for possible courses of action at a given time.

The choices for user interface devices include soft function keys, light pens, joystick, trackballs, voice input/output, touch panels, keyboards, valuators, tablets, multiple high-resolution (1024 line) color graphic displays, hard-color copy camera units, etc.

CONCEPT VALIDATION

A formal requirement specification was published in the 1980 TRW Man-Machine Interface IR&D Final Report (10). The report was sent to internal reviewers and outside consultants. To further refine the concepts and gather additional ideas about the implementation of the DDL, works mentioned in references 5 through 12 were studied. Relevant ideas and techniques from these publications were considered.

SYSTEM DESCRIPTION - FLAIR: CONCEPT

FLAIR (Functional Language Articulated Interactive Resource) is a voice menu-driven Dialog Design Language which embodies all relevant hardware and software in the testbed. Through use of a variety of input devices (such as tablet, light pen, trackball, and voice recognition) FLAIR controls a hierarchy of user command menus. To date, actual FLAIR commands are entered via voice recognition, but the other devices can be used for user's menu selection and coordinate location. Additionally, the keyboard is used to enter long strings of information, such as labels and file names.

FLAIR CAPABILITIES

FLAIR provides the user the basic services to manipulate graphics objects or primitives using ACM core standard graphics. These include:

-- drawing of lines, boxes, circles; writing of text in stroke or hardware character sets, in any size or orientation; selection of colors (including blinking configurations); fill of any enclosed area by a color; hardware pan and zoom of the screen; read or write a pixel image to/from the screens; erasure of entire display (or single item); freedrawn lines controlled by the cursor movement.

HIGH-LEVEL FEATURES

In addition to the basic features, FLAIR provides macro facilities to perform the following by voice commands:

-- construct a shell for the FLAIR user to define and control a menu hierarchy of his own design in his operator's dialog; access to a world-wide geographic data base. By describing two points to form a rectangular box, a geographic outline on the area can be displayed with progressive detail; generate full color dynamic barcharts and piecharts; construct new symbols from existing symbol building blocks; compute arithmetic expressions by voice entry. This calculator is complete with six "registers."; access the relational data base for graphical entity attribute storage and retrieval; generate a grid system which is used to perform precise cursor positionings for lines, charts, and text; access the voice synthesizer unit.

USER INTERFACE

The user interface utilizes relevant input/output devices for any action. The two screens in the MMI workstation work in tandem to prompt and aid the users. The human voice is currently used to input all commands which manipulate FLAIR. Urgent or exceptional messages are announced by the voice synthesizer. Pick or point actions are handled by the graphics tablet or comparable input devices.

SCREEN LAYOUT

FLAIR currently makes use of six separate graphical "windows" located on the two workstation monitor screens. These are (Figure 6):

a) The Auxiliary Information Window, which is reserved for large-scale display of the symbol menu as well as the textstyle and linestyle menus. This window covers the entire 19" monochrome screen.

b) The Primary Display Window, which contains all user-defined graphical data (such as lines, barcharts, maps, and user-built menus). It can be thought of as the "window" into the user's world.

c) The Calculator Result Window, which contains all intermediate and final results from FLAIR's voice-activated desk calculator.

d) The Error/Information Window,which is used for display of messages to the user, including error messages. This window only contains relevant text if the message is current.

e) The Dynamic Command Window, which contains, at any given time, the list of words which the user can "say" to FLAIR for it to perform an operation. The command lists are dynamic because the meanings of the commands are accepted for only the allowable actions at that particular sequence. As the command is accepted, the list changes or refreshes after acknowledgement of receipt of a word.

f) The Current Color Window, which graphically portrays the current color. It also contains an hourglass-shaped symbol, which appears when the system is too busy to accept input at that time.

INPUT TO FLAIR

As previously stated, FLAIR itself is primarily controlled by voice commands. In addition, the operator uses the graphics tablet for graphic primitive location and graphical picks. When it comes to operator-designed scenarios (menu hierarchies), the operator has his choice of joystick, light pen, trackball, or graphics tablet for menu item picks.

VOICE SYNTHESIS

The voice synthesis unit is used by FLAIR to flag operator errors and major events, to enable or disable certain FLAIR modes, and to prompt the operator in tandem with messages in the Error Information Window. The operator can also utilize the voice unit in his dynamic scenarios.

COMMAND HIERARCHY

FLAIR is controlled via voice commands, which select paths from a tree-of-action items. Every command node may lead to another branch or conclude with a leaf. In general, the top command level allows selection of an action (verb), level two, an object (noun), and level three, a modifier (adjective or noun). Regardless of context and depth in the tree, a set of global words is constantly monitored by FLAIR. These words (and some phrases) control overall FLAIR operation and voice recognition performance. These are....ABORT..STOP..FINISHED..RESET..READ THRESHOLD..SET THRESHOLD.

FLAIR COMMAND SYNTAX DESCRIPTION

At the topmost (root) level of the FLAIR syntax, there currently exist fifteen root words which, in turn, activate the other 85 commands to accomplish a task. These root words are:

MAKE - allows construction of barcharts, piecharts, and composite symbols

SAVE - (a) saves the Primary Display Window as a digital image, and (b) initiates/terminates construction of a frame or scenario

ERASE - allows erasure of selected elements, or the entire Primary Display Window

DRAW - enables graphic primitive construction

COMPUTE - engages the voice-activated calculator

GEOGRAPHICS - allows definition and display of some portion of the world, with or without a longitude/latitude grid.

ZOOM - enables hardware zoom and pan features to enlarge and scan features on the screen

VOICE OUTPUT - prompts for line of text, then attempts to have voice synthesis unit pronounce it

COLOR - allows selection of current color

QUERY - allows user to query the data base attribute of a specific symbol

UNZOOM - resets zoom factor and pan position to normal modes

CONTROL - (available only if frame construction mode is enabled) permits construction of saved static frames and dynamic scenarios

FILL - allows user to specify the objects he would like filled, or to enable default fill mode

LIGHT PEN - enables light pen (and other RAMTEK devices) for user-defined menu picking. It automatically disables the tablet at time of pick

SYMBOL BUILDING BLOCK DEFINITION

Contained with FLAIR is a menu to access 30 basic symbols. The menu contains rectangles, circles, crosses, radar disks, etc., and can be drawn on the screen by themselves or combined to form composite symbols. These composite symbols, once formed, are independent of the basic symbols, and can be used as new symbols. Definitions of the 30-symbol building blocks are kept in the FLAIR data base. However, there are another 120 basic symbols that FLAIR can access via an offline process through the DBMS update utilities.

GRAPHICS - DATA BASE BINDING

FLAIR provides the capability to manipulate user-defined data, and associate it with graphical items portrayed on the screen. In this way, he can query the item in some fashion to retrieve all or part of its data.

FLAIR provides graphics-data base binding schema using the INGRES DBMS. Any symbol, whether it be a building block or composite, can be associated with data base 'attribute' - in this case, a string of text. If, at some later time, the operator wishes to inquire as to the string with a particular symbol, he can pick that symbol from the primary display window with the tablet. The corresponding attribute (if any) is then displayed in the message area.

Future implementations will allow an operator to not only associate symbols with more complicated data base structures, but to define these structures as needed.

DYNAMIC SCENARIO CONSTRUCTION AND PLAYBACK

When the operator wishes to construct a dynamic scenario, he enables what is known as "save" mode. From that point on, all FLAIR commands and locator input are recorded under an operator-specified scenario file. The actual scenario file is the list of FLAIR commands stored in the sequence specified. If an error occurs during construction or the user wants to update the scenario, these files can be edited with a regular text editor. To "play back" a scenario file, the operator then enters "display file" mode.

FLAIR not only can create, store and retrieve static frames, it can allow the user to define and control a hierarchy of menus. This is the feature which makes FLAIR a dynamic scenario generator. Through the definition of the user's menus, an operator dialog can be constructed. A client can interact, as he traverses through the selection of the menus, with the sequence and the path of the scenario controlled totally by the client.

Individual menu items are defined on the screen as pick-sensitive boxes or circles. If an operator "picks" within the boundaries of a menu item, a previously defined action, as specified by the writer of the operator dialog, takes place.

CONCLUSION

We have presented the current configuration and the functions of a Dialog Design Language (FLAIR) developed by TRW. The language combines the power of ACM Core-standard computer graphics, a relational data base management system, voice recognition, and MMI input/output devices to form a tool for system designers. This tool will enable system designers to realize and manipulate conceptualized operator dialog. In this way, various approaches to the design can be tried so that the best concept or a combination of concepts will be implemented in the real system.

FLAIR has been in operation since August 1981. The preliminary results are very promising. To generate a frame with functional menus takes very little time. One typical scenario which contained maps, symbols with attributes, various colors, a maximum of six separate functional menu selections, and a series of five hierarchy levels took three days to complete. Based on experience, if the same scenario had been done using traditional programming, it would have taken at least one and one-half to two weeks.

FUTURE PLANS

We intend to develop more advanced features for FLAIR, such as a multi-tasking DDL. In this way, under the same execution node, multiple workstations can share the logic and provide multi-workstation communications to simulate two or more workstations communicating with each other.

Another major task will be to provide an interface for the user's own algorithms. This shell will provide for external simulations not easily or necessarily incorporated into FLAIR. In this way, the user need not gain access to the internals of FLAIR.

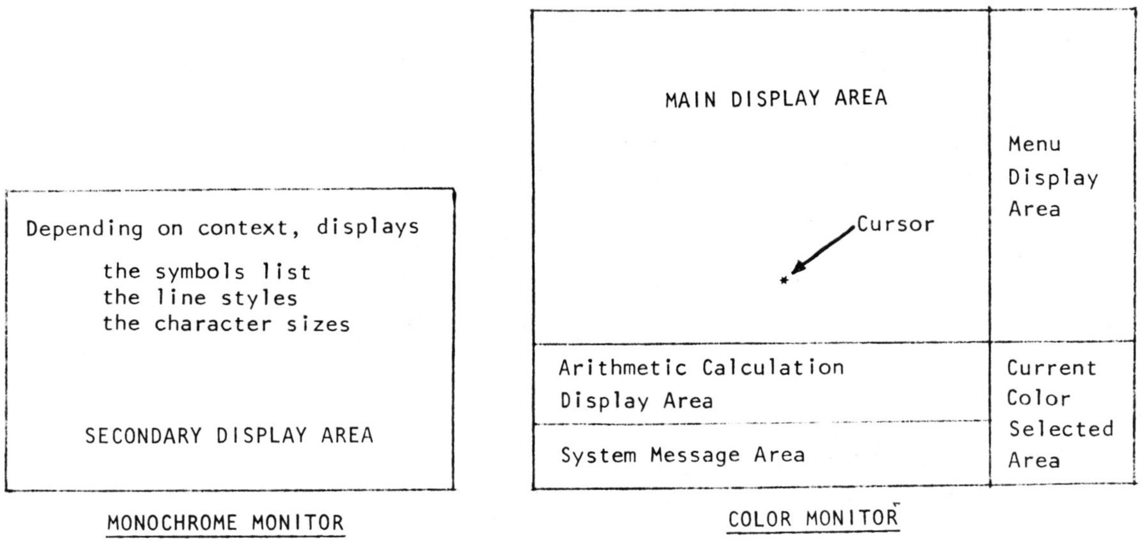

Figure 6. FLAIR Window Areas

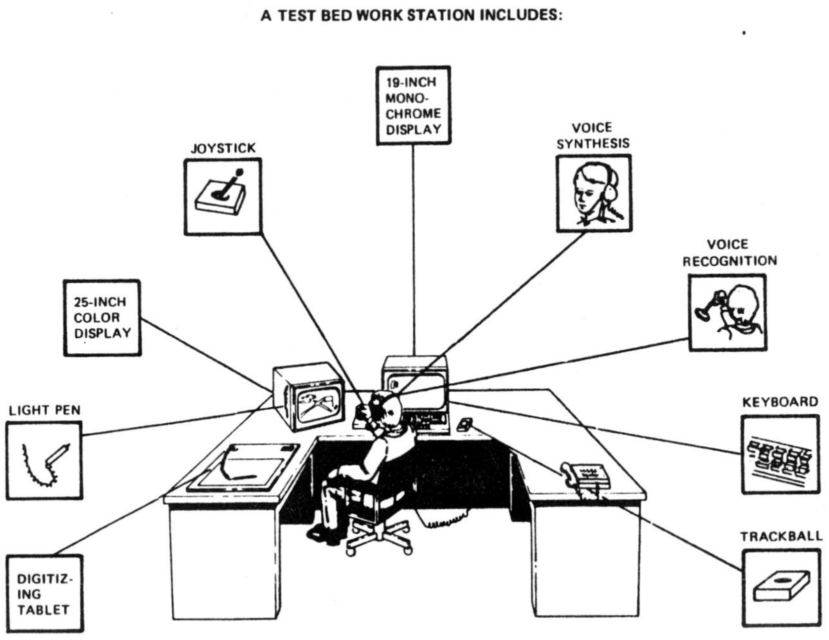

Figure 1.

REFERENCES:

1) Bergeron, R.D., Bono, Peter, Foley, James, "Graphics Programming Using the Core System." Computing Surveys Dec 1978

2) Newman, W., Sproull, R., "Principles of Interactive Computer Graphics," McGraw-Hill, 2nd Edition 1979

3) Stonebraker, M., Held, G., Wong, E., "INGRES - A Relational Data Base System," Proc. AFIPS, Volume 44

4) ACM-SIGGRAPH, "Status Report of the Graphics Standards Planning Committee," Computer Graphics, Vol. 13, No. 3, August 1979

5) Treu, Siegfried, "A Framework of Characteristics Applicable to Graphical User-Computer Interaction," P61-71 User-Oriented Design of Interactive Graphics Systems. Based on ACM/SIGGRAPH Workshop Oct 14-15, 1976, 1977

6) Smith, Sidney, "Man-Machine Interface (MMI) Requirements Definition and Design Guidelines - A Progress Report" P39-81, ESD-TR-81-113 Feb '81

7) Foley, J.D., Chan, P., Wallace, V.L., "The Human Factors of Graphic Interaction: Tasks and Techniques," Tech.Rep. 508, US Army Research Institute for the Behavioral and Social Science, Sept. 1981

8) Bolt, R., "Put-That-There Voice Gesture at the Graphics Interface," ACM/SIGGRAPH '80 Conference Proceedings, July 1980

9) Parsons, H., "Man-Machine System Experiments," The Johns Hopkins Press, 1972

10) Deaton, B., "Dialog Design Language Functional Requirements Specification," TRW 1980 Man-Machine Interface IR&D Final Report, Dec '80

11) Kloster, G.V., "Draft and Guidelines for the Development of the MMI Design Guidebook," TRW 1980 Man-Machine Interface IR&D Final Report, Nov 1980

12) Hanau, P., Lenorovitz, D., "Prototyping and Simulation Tools for the User Computer Dialogue Design; ACM/SIGGRAPH '80 Conference Proceedings, July 1980