

Image Cover Sheet

CLASSIFICATION

UNCLASSIFIED

SYSTEM NUMBER

504207



TITLE

CAD INTERFACE DEVELOPMENT FOR LOCATE WORKSPACE LAYOUT TOOL

System Number:

Patron Number:

Requester:

Notes:

DSIS Use only:

Deliver to:

Report Documentation Page				Form Approved OMB No. 0704-0188	
Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.					
1. REPORT DATE MAR 1997		2. REPORT TYPE		3. DATES COVERED 00-00-1997 to 00-00-1997	
4. TITLE AND SUBTITLE CAD interface Development for WCATE Workspace Layout Tool				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S)				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Artificial Intelligence Management and Development Cooperation,206 Keewatin Avenue,Toronto, ON M4P 1Z8 Canada,				8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)				10. SPONSOR/MONITOR'S ACRONYM(S)	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited					
13. SUPPLEMENTARY NOTES					
14. ABSTRACT This report describes work that extends earlier efforts at building a graphical user interface for LOCATE, a workspace layout design tool used to assess the efficiency of communication among entities in a workspace. The primary objectives of the current work were to add the following functionality to LOCATE: a) an interface between LOCATE and other CAD packages; b) standard editing functions; c) workspace boundary tools for wall door and window; c) saving and retrieval of generic objects (chairs, desks, cabinets, etc. including the workspace boundary objects just mentioned) and their attributes; d) capabilities for multiple selection and grouping; and e) a variety of minor refmemements. Beyond the stated requirements, a facility was added for automatic grouping of generic objects within workstations and obstructions. That facility eliminates the need to group and ungroup objects repeatedly designs are developed. A notion of layered objects in the interface evolved concurrent with the work on automatic grouping, and three layers were identified. Selection, grouping and relative positioning of objects in the interface now are done within those layers. A variety of checks were added to LOCATE that warn users when they are about to violate the constraints of the interface and others that make sure that they have entered the data necessary to the running of a cost function, LOCATE's measure of communication efficiency. Finally, suggestions were made that could serve as a focus for future development work.					
15. SUBJECT TERMS					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT Same as Report (SAR)	18. NUMBER OF PAGES 54	19a. NAME OF RESPONSIBLE PERSON
a. REPORT unclassified	b. ABSTRACT unclassified	c. THIS PAGE unclassified			



Artificial Intelligence
Management and Development Corporation

Prepared for: Public Works and Government Services
Prepared by: Artificial Intelligence Management and Development Corporation
AIM (AC182, March, 1997)
Contract: W7711-6-7322/001/SRV
"CAD interface Development for *LOCATE* Workspace Layout Tool"

Scientific Authority:

Mr. Keith C. Hendy
Human Engineering Section
Defence and Civil Institute of Environmental Medicine

© 1997 Her Majesty the Queen in right of Canada, as represented by the Minister of National Defence

**CAD
Interface Development
for *LOCATE*
Workspace Layout Tool
- Final Report -**



Artificial Intelligence
Management and Development Corporation
206 Keewatin Ave., Toronto, Ontario M4P 1Z8

Table of Contents

Abstract	iv
Background	1
Research Approach	
Objectives	2
Development of the CAD Interface	
Previous Work	4
Current Work	6
Refinements to LOCATE	
Saving Objects and Their Attributes.....	9
Automatic Grouping of Objects.....	10
Layering of Objects	12
Relationships Among Objects Within Layers	13
Standard Edit Functions.....	14
Workspace Boundary Tools	16
Multiple Selection and Grouping.....	16
LOCATE's Verification Checks.....	17
Minor Refinements	18
Additional Refinements During the Contract Period.....	22
Summary	23
Next Steps	
Feature Extensions	26
System Checks	32
A Common Interface	33
Other Development Work.....	33
References	35

Appendices

Appendix A: Annotated List of Source Code Files for the LOCATE Interface:	A-1
Macintosh Implementation	
Appendix B: Additional Functional Requirements, Problems and Bug Fixes	A-6

Figures

Figure 1: The Cost Function History Window	20
--	----

Abstract

This report describes work that extends earlier efforts at building a graphical user interface for LOCATE, a workspace layout design tool used to assess the efficiency of communication among entities in a workspace. The primary objectives of the current work were to add the following functionality to LOCATE: a) an interface between LOCATE and other CAD packages; b) standard editing functions; c) workspace boundary tools for wall, door and window; c) saving and retrieval of generic objects (chairs, desks, cabinets, etc., including the workspace boundary objects just mentioned) and their attributes; d) capabilities for multiple selection and grouping; and e) a variety of minor refinements.

Beyond the stated requirements, a facility was added for automatic grouping of generic objects within workstations and obstructions. That facility eliminates the need to group and ungroup objects repeatedly designs are developed. A notion of layered objects in the interface evolved concurrent with the work on automatic grouping, and three layers were identified. Selection, grouping and relative positioning of objects in the interface now are done within those layers.

A variety of checks were added to LOCATE that warn users when they are about to violate the constraints of the interface and others that make sure that they have entered the data necessary to the running of a cost function, LOCATE's measure of communication efficiency. Finally, suggestions were made that could serve as a focus for future development work.

Background

LOCATE is a computer aid for design, developed for workspace layout but applicable to a variety of other areas (Hendy, 1984; 1989; Hendy, Berger, & Wong, 1989). The program supports communication by vision, audition, touch and movement. It allows designers to input information about location, orientation and structure of elemental workstations in a workspace layout and to generate cost functions as indicators of the efficiency of communication in a workspace. Different configurations may be tried, and their cost functions compared, as a way of helping a user determine the best design.

LOCATE was originally written in DEC VAX Fortran 77, with labour intensive keyboard data input. The LOCATE software subsequently was enhanced with a Graphical User Interface (GUI), using Neuron Data's Smart Elements software tools. The GUI version of LOCATE currently runs on a Macintosh computer (the SGI Iris Indigo class of machine being its ultimate target platform) and generates output files readable only by LOCATE itself. That is, it does not presently support the sharing of files with other CAD packages, like AutoCAD for example. This contract adds further functionality to LOCATE's GUI and provides for exporting and importing files to and from other CAD packages.

Previous work for DCIEM (W7711-5-7268; W7711-0-7119), conducted by Artificial Intelligence Management and Development Corporation (AIM), added basic elements to the LOCATE interface. Other on-going work (W7711-6-7321) is adding a help system with intelligent aiding to LOCATE's functionality.

Research Approach

Previous work on LOCATE served as a model for the approach taken in the current contract, with the principal goal being to design and build an interface between LOCATE and other CAD software such as AutoCAD. A secondary goal was to add to and refine other areas of LOCATE's functionality.

Objectives

The main objectives of the present study were:

- to develop an interface between LOCATE and other CAD applications, such as AutoCAD, that supports importing and exporting files in the industry standard, .dxf file format.

As Locate produces considerably more information about its specialised objects than might normally be stored as part of typical CAD drawings, special emphasis will focus on ways of retaining that information in exported files so that the relevant portions may be usable even after export and modification within other CAD applications.

- to provide support for editing of LOCATE objects, namely, the ability to Cut, Copy, and Paste those objects in a workspace.
- to extend LOCATE's tool palette so that users may add Workspace Boundary objects in a workspace, including at least the following components: walls, doors, and windows.
- to add functionality that allows users to save workspace objects and their attributes.
- to provide for the multiple selection and grouping of objects in support of other functionality mentioned above, such as editing and saving of customised objects to an object library.
- to provide for other, minor refinements to LOCATE that include, but are not be limited to, the following:
 - adding "Reset" buttons to the Priority Weights and Link Functions Windows;
 - cleaning up multiple screen refreshes, as appropriate;

- drawing elliptical obstructions as ellipses instead of as circles;
- locking of buttons in the cost function window to stabilise their display;
- making cost functions, as displayed in the cost functions window, cumulative within and across sessions;
- expanding the “All Objects Info Window” to include all instances present in the workspace for the type of object selected;
- modifying LOCATE’s design window so that the title is the name of the design;
- adding the Length-to-Breath attribute for Fixed Obstructions;

Development of the CAD Interface

Previous Work

Earlier work (W7711-5-7268) examined how LOCATE might interface to such popular CAD programs as AutoCAD. Links between LOCATE and those CAD packages are useful in that a designer is not required to re-create the basic elements of a design when moving from one environment to another. Of course, it is not expected that all information created in any one environment will port to another.

For example, information about the kinds of links that exist among workstations, link function data, obstruction data, and other information saved in LOCATE files likely will not be readable by other CAD packages. Nor will all the information saved, say, in an AutoCAD file be readable necessarily by LOCATE.

The information stored in commonly readable file formats will include basic design objects like standard geometric shapes, lines, grouped items and text. Information may be included in the various files that may be readable by some software but not by others.

In considering the requirements for developing an interface between AutoCAD and LOCATE, the focus has been on using standard .dxf (Data Exchange Format) text files. In considering the requirements for developing such an interface, it was determined that working with a complete version of AutoCAD was neither necessary nor advisable.

The reasons include the fact that Autodesk, the developers of AutoCAD, have no plans to release a new version for the Macintosh, the current development platform for LOCATE. Also, since LOCATE is a 2-D environment, for which the full 3-D capabilities of AutoCAD are not required, and since a 2-D version of AutoCAD is available for the PC and finally, since .dxf text files are easily transferred among different hardware platforms, a 2-D, PC version of AutoCAD called, "AutoCAD LT", was acquired as an initial step in building the LOCATE/AutoCAD interface.

The characteristics of the .dxf files, which are produced by AutoCAD LT, were examined in some detail. There are four sections to a .dxf file:

- Headers:

The Header section contains roughly 140 separate items, their group codes, variable types and values.

- Tables:

The tables section contains summary information about certain aspects of the design environment such as the coordinate system, the view, the viewport, layers, styles and applications registered with the drawing;

- Blocks:

This section contains references to objects that make up the “Blocks” used in a drawing. They can be simple or complex user-defined items, which, in AutoCAD, can be inserted into a drawing at any point. External Reference (Xrefs) are also included here.

The difference between a Block and an External Reference is that when a change is made to the latter, instances of that item are automatically updated in all files that reference it.;

- Entities:

The final section is the Entities section, which includes items a user draws in a design space, identified by name that appear along with various group codes, variable types and values.

AutoCAD represents some items at low-levels so that they may be represented and manipulated in other drawing environments at varying levels of abstraction. For example, a rectangle is written out to an AutoCAD .dxf file as a polyline with four vertices.

It was determined that writing a utility program for LOCATE that would be capable of reading .dxf files, produced by AutoCAD or other CAD programs, would be reasonably

straightforward. The .dxf format makes it easy to ignore information not needed while reading in information that is needed. For example, reading information from a 3-D AutoCAD program simply involves ignoring z-axis data.

Just as reading .dxf files, it should be relatively straightforward to represent complex items in LOCATE such as a prototype “Block” for an Elemental Workstation and its component parts that can be read by AutoCAD or used in other design environments. A similar utility will be required to write out information from LOCATE in the .dxf format. Both the reading and writing utility programs can be incorporated as Import/Export commands in LOCATE.

One final note is that since AutoCAD is an evolving program, variables, codes, etc. are being added and modified in .dxf files on an on-going basis. Thus, the import and export commands in LOCATE should be revisited periodically to take advantage of features available in new releases of AutoCAD or in similar updates to the LOCATE interface itself.

Current Work

Work during the current contract focused on developing basic requirements for importing and exporting files from LOCATE and AutoCAD. Having explored the structure of .dxf files, as described in the previous section, code was written to allow LOCATE to generate “Block” definitions for its basic units. Those included a(n):

- Elemental Workstation
- Source/Receiver Node
- Elemental Obstruction
 - Rectilinear
 - Elliptical
- Fixed Obstruction
 - Rectilinear
 - Elliptical

Several instances of those object types were created in the Macintosh implementation of LOCATE and exported in a .dxf file format. The resulting text file was converted to a PC format and opened in the PC version of AutoCAD LT. The Block definitions of the LOCATE prototypes were verified and it was possible to create new instances from the various Block definitions in the AutoCAD design window.

As various attributes were added to the objects of the LOCATE interface, they also were added to LOCATE's .dxf file Block definitions. Among those attributes were the name of the object, the object number and the contents of the object. Further work verified that those attributes appeared in the Block definitions when translated and opened in AutoCAD.

Concurrently with developing a capability for generating .dxf files from LOCATE, other, similar work was done to allow LOCATE to read .dxf files created in AutoCAD. In addition to reading in new object instances created in AutoCAD from LOCATE Block definitions, a variety of simple objects (rectangles, circles, arcs, etc.) may now be exported in a .dxf file format and read directly into LOCATE. What cannot be read into LOCATE are objects created using Block definitions other than those of the LOCATE objects that are created in AutoCAD. That capability should be explored in future development efforts.

At this point, it was necessary to set aside the work on the CAD interface while other work proceeded with LOCATE—work that incorporated other objects and their attributes into the LOCATE interface.

Several new object types and their attributes were added to LOCATE, including the workspace boundary objects of wall, window and door, and the generic object types: chair, desk, console, computer, cabinet, partition, stairway, etc.. Other development work that preceded a return to the CAD interface effort included the development of an automatic grouping facility for workstations and obstructions and a method of recalling previous design configurations for which cost functions had been run, the latter was work done under separate contract (W7711-6-7320) in the context of a LOCATE usability study.

Following the implementation and debugging of those features, attention once again returned to the CAD interface. The new object types and their attributes are now part of the Block definitions exported in each LOCATE .dxf file. Further refinements to the import command now allow LOCATE to recognise and use those aspects of .dxf files generated by other software packages such as AutoCAD while ignoring information that it does not need or cannot use in its representations.

Whereas object types, their attributes and instances can now be exported in a standard .dxf file format what would prove useful in future is to have the complex arrays of information represented by Function and Priority Weight data exported as well. Those data would likely find little use by designers in AutoCAD but would be useful if they were to import a file from LOCATE, modify portions of the design in AutoCAD and then export that back for further use in LOCATE.

Of course, in that case, objects from the original LOCATE file may have been deleted or modified in AutoCAD in a way that would make some of the complex array data useless. Ways of detecting such changes would be needed but could save the LOCATE designer considerable time and effort in working with a design originally created in LOCATE and modified in AutoCAD, or some other design package. It would also considerably enhance the flexibility of data sharing among LOCATE designers and those using other CAD packages.

Refinements to LOCATE

With the exception of the minor refinements listed at the end of this section, the list of improvements to LOCATE given below reflects the order in which they were addressed.

Saving Objects and Their Attributes

A major drawback of the previous version of LOCATE was the fact that although all the ingredients were present to create a design and run cost functions, the generic objects that designers normally create in a workspace could only be created but not saved. That is, generic items like chairs, desks, computers, cabinets and the like could be created but were not saved along with the design. The design could be saved, recalled and cost functions run but no objects that defined the various obstructions in the workspace would be displayed.

In addition, the generic objects, like other objects in the workspace, were limited in terms of the attributes that could be used to identify them. Also, there was limited flexibility in a users ability to size, position and rotate generic objects.

Development in this phase has produced a set of generic objects that can be manipulated in a variety of ways, including saving and retrieving, cutting, copying and pasting, resizing and rotating both in their attributes windows and, dynamically, by clicking, dragging, and rotating.

A special grouping feature, to be explained in the next section, provides users further flexibility with generic objects by allowing them to be placed at a variety of points in the workspace, namely, inside elemental workstations, inside elemental obstructions, which are inside those workstations, inside fixed obstructions, and out in the workspace, by themselves. When placing generic objects inside containers, i. e., workstations or obstructions, LOCATE automatically groups these objects in a part relation to the container within which they are placed.

In the current implementation, generic objects are tagged to the global workspace coordinate system, that is, their positions are defined relative to the coordinates of the workspace regardless of where they reside. This was done because they, in fact, can appear in a variety of contexts. Future consideration should revisit the coordinate system to which they, and all other objects, should be anchored. For purposes of the LOCATE analysis, the

issue of anchoring is clear, but in the interest of a designer's ease of use of LOCATE, different decisions may be needed.

Automatic Grouping of Objects

Just as LOCATE provides a user with the automatic creation of links between any new workstation placed in the workspace and every other workstation that is already there, work was done on an automatic grouping facility that would eliminate the need for the repeated use of grouping and ungrouping commands. A need for this feature had not been part of the original specifications for the contract and only emerged as the work proceeded.

Early into the contract, it became clear that several different types of object seemed to naturally function as units. Specifically, it seemed reasonable to assume that objects such as chairs and desks typically appear inside workstations. Although some of those objects also appear outside workstations in the workspace, when there, also often appear as part of fixed obstructions.

It seemed reasonable, therefore, that designers using LOCATE might spend considerable time grouping and ungrouping objects as their designs evolve. To ease that burden, it was decided that generic objects would be grouped automatically when placed inside various LOCATE "containers." Those containers were identified as workstations, elemental obstructions and fixed obstructions.

Whenever a user creates a generic object and places it inside one of those containers, the object now automatically becomes part of that container. If the container is moved or rotated the object moves and rotates with the container. Objects placed within elemental obstructions illustrate the case of an object that is inside a container, which is inside another container. Since elemental obstructions appear inside workstations, and both are container objects, when a chair or desk, say, is placed inside an elemental obstruction, it is inside a container in a container.

If any portion of the workstation, outside the obstruction, is selected and moved or rotated, the workstation and all its contents move and rotate at the same time. If any portion of the obstruction is selected and moved or rotated, the obstruction and all its contents move and rotate at the same time. The workstation, of course, does not move or rotate.

If other generic objects such as cabinets, partitions, columns and the like are placed inside fixed obstructions, they are automatically grouped with those obstructions and move and rotate with them.

Another way users may take advantage of this automatic grouping facility in LOCATE is by populating a workstation with desks, chairs, consoles, computers, etc., and then dragging out an obstruction around those objects to identify them as an obstruction. The objects become part of the obstruction. In a similar way, users may populate the workspace and then drag out workstations around those objects and then drag out obstructions around the objects inside the workstation objects.

This use of automatic grouping only happens when the obstruction or workstation is first dragged out. If a user moves an existing obstruction over the top of other objects in the workspace, they do not become part of that obstruction. On the other hand, if the user drags an object inside that existing obstruction it will become part of it. This same process also applies to objects and fixed obstructions.

Thus, the automatic grouping facility provides considerable flexibility to users in terms of the order in which they create their designs.

A problem that arose after adding the automatic grouping feature to LOCATE was that users found that as long as they grabbed the edges of an obstruction and moved it that all objects inside the obstruction would move with it. If they grabbed one of the objects inside the obstruction, however, that object moved but not the obstruction.

The reason the grouping had been organised in that way was to allow the user to move items in and out of obstructions as desired. Although the ability to do just that was needed, it proved to be disconcerting when chairs, desks, computers, etc. had been placed into positions where the user wished them to remain. When the user had positioned objects in that way, it became increasingly annoying to have them move when accidentally selected, when the user was simply trying to move the obstruction or the workstation within which they had been placed. That meant wasted time for the user in repositioning the object.

Consequently, it was decided that once objects are placed inside obstructions or workstations that selecting any part of the obstruction or workstation will select the obstruction or workstation only and not the objects contained within. Thus, a user may now grab and move a chair, say, that is sitting inside an obstruction and the whole obstruction will move.

Of course, that solved only part of the problem. There still needed to be a way for a user to select generic objects and move and rotate them. The solution adopted was to require the user to hold down the Control (Ctrl) key, select the object and then move, resize or rotate it. In moving the object, the user may move it anywhere within an obstruction or workspace, outside of an obstruction and into the workstation, or even outside into the workspace and into a fixed obstruction.

With regard to workstations, this logic also applies to the Source/Receiver (S/R) node. If an S/R node is grabbed and moved, the entire workstation and all its contents move with it. In order to move or rotate an S/R node only, the user holds down the Control key and then selects and moves or rotates the node. S/R nodes cannot be resized since size is irrelevant and they cannot be moved outside of workstations as they are defined solely in terms of the workstations within which they appear.

Some concern still remains regarding the automatic grouping facility and whether users will find it as useful as it appears, or whether they will prefer the standard, manual grouping. In a recent study (W7711-6-7320) on the usability of LOCATE, one of the subjects was particularly impressed with the automatic grouping facility, expressing the opinion that it saved him considerable effort by eliminating the need to do it himself.

More experience with LOCATE will clarify how useful this feature is and how it will interact with a standard grouping facility that will also be part of the LOCATE application.

Layering of Objects

Concurrently with the development of the automatic grouping facility another feature that seemed consistent with the structure of LOCATE was a layering of its different types of object. Three layers of object types were identified: 1) at the lowest layer are workstations; 2) the middle layer contains elemental and fixed obstructions; and, 3) the top layer is composed of generic objects, such as chairs, desks, cabinets and the like.

Generic objects as a group may be thought of as being the “closest” to the user. Objects in that category will obscure other generic objects, elemental and fixed obstructions and workstations. Instances of obstructions will obscure each other and workstations but not generic objects. Workstations will obscure only other workstations.

No doubt, adjustments to the concept of layering will be required in future as other notions, like that of elemental and fixed obstructions and of automatic grouping are refined. A current concern, for example, is how to display an obstruction in such a way that users understand that the object or objects that make them up constitute the obstruction, but do not infer that the obstruction has an existence separate from those objects.

One solution for how best to represent obstructions would be to add several pixels of shading or colour around the border of the object(s) or the part of an object that constitute(s) the obstruction. In that way, the boundary pattern or colour would be detectable but would not extend so far beyond the edges of its constituents so as to imply to users that it was an object with its own separate existence.

Presently, when a user wants to identify something as an obstruction, he selects the appropriate obstruction tool from the palette and drags out a rectangular or elliptical obstruction in the workspace. In doing that, the boundaries of the obstruction almost always extend well beyond the objects that are intended to make up the obstruction.

That type of display does give a user the sense that an obstruction has a boundary that extends well out from the boundary of the object(s) that actually constitute the obstruction. One solution is that, after dragging out the obstruction, to have the boundary “snap-to” the boundary or boundaries of its constituent object(s).

Alternatively, the “Minimize Boundary” command in the Execute Menu, which has been used to minimise workstation boundaries could be extended to apply to obstructions. Thus, an obstruction would maintain the same boundary as when the user created it, but would “snap-to” the boundary of its constituent object(s) when the user selects the “Minimize Boundary” command (this would require one to represent the new boundaries mathematically in the form of an attenuation function - not a trivial task).

Relationships Among Objects Within Layers

Layering defines relationships among categories of object types. There also is a need to specify the relationships among objects within those categories. That need is met by the menu commands, “Bring Forward”, “Send Backward”, “Bring to Front” and “Send to Back.” As part of the current work, those commands were added to the Align Menu and allow users to define relationships among objects within their respective layers. For example,

workstations that overlap other workstations may be re-ordered relative to one another, so that when a user clicks on overlapping objects, he may define which one will be selected. In a similar way, obstructions may be re-ordered relative to other obstructions and, within the “topmost” layer, generic objects may be re-ordered with respect to other generic objects.

The adjustment to the automatic grouping function discussed earlier, in which to access generic objects a user must hold down the Control Key, has produced a qualification on the way in which objects are accessed in the interface.

As will be recalled, it was found that once generic objects were placed in a container, e.g., a workstation or an obstruction, and thus automatically grouped with that container, that selecting and moving is generally understood by users to imply selecting and moving their containers and not the individual objects themselves.

Consequently, when a user clicks, holds and drags an object in, say, an elemental obstruction, the expectation is that the obstruction and all its contents will move, not just the individual generic object that is part of the obstruction. The LOCATE interface was modified to accommodate that assumption by users.

Of course, generic objects themselves may be moved, but to do so the user must hold down the Control key while clicking and dragging or when selecting a generic object for rotation. In that way, the user does not accidentally alter a part of an obstruction out of the fixed position or orientation in which he has placed it.

All that this qualification means, as far as the re-ordering of objects within layers, is that re-ordering at the topmost layer can only be done by holding down the Control Key and then selecting the object that one wants to re-order. Once the object is selected, any of the menu commands, “Bring Forward”, “Send Backward”, “Bring to Front” or “Send to Back.” may be chosen.

Standard Edit Functions

Work on editing functions included providing support for the standard Cut, Copy, Paste and Duplicate functions; the Clear function had been available for some time. Also available had been an ability to cut, copy and paste text within and across windows and dialogue boxes.

Work in the area of editing focused on allowing users to select various objects in LOCATE and to copy and paste them into other parts of the workspace. This was more difficult than at first appeared since many of the objects, i.e., workstations and obstructions, contain other objects.

Also, since workstations are linked to other workstations and contain function and argument data, the decision as to which of those elements should be copied and pasted was not immediately apparent. For example, what relation should a newly pasted workstation have to the workstation from which it was copied.

The solution adopted was to copy size and rotation information along with the link function and argument data. What is not copied and pasted is priority weight data from the original workstation.

Positioning of pasted workstations is down and to the right of the original, at an offset slightly greater than the original's diameter. That means pasted workstations appear completely outside of the original workstation, with their boundaries touching.

A user may specify where an object is to be pasted and that is true of workstations as well as other objects. That means that a user may override the positioning of a workstation and click inside of the original before pasting. The newly pasted workstation is centred at the point where the user clicked.

Workstations may also contain obstructions and those are copied and pasted along with the workstation. All function and argument data from the original obstruction are copied and pasted with the new obstruction.

With regard to copying and pasting elemental obstructions **only**, they and their contents are copied and pasted, by default, inside the workstation at an offset to the original equal to half the diagonal of the obstruction's bounding rectangle in an unrotated state.

If a user identifies a location, which is in the workspace but outside the workstation, where he would like to paste an elemental obstruction, an alert appears telling the user that elemental obstructions may only be placed inside of workstations. If future development eliminates the need for some of the distinctions currently made among obstructions, this restriction may be eliminated.

Again, regarding the placement of objects in general, users may indicate either a location where they would like an object pasted or accept the default offset determined by the system.

Applying standard edit functions to fixed obstructions works in a similar way to the way they are applied to elemental obstructions. All objects within a fixed obstruction are copied and the obstruction is pasted at an offset to its original.

Finally, copying and pasting generic objects like chairs, desks, computers and the like may be done anywhere in the workspace. If a user does not identify a location where the object is to be pasted, the object is placed at an offset to the original object from which it was copied.

Workspace Boundary Tools

Three types of workspace boundary tool were implemented as part of the current contract: 1) wall; 2) window; and, 3) door. Representations for those objects were part of the tool palette but their functionality had not been added before this contract.

In addition to adding the boundary tools, an offset of 10% was added to the workspace dimensions so that users could comfortably place those boundary objects either inside or outside the actual boundaries of the workspace. Also, the workspace boundary is clearly identified for user by the addition of a blue border.

Multiple Selection and Grouping

Work on grouping, discussed in a previous section, pointed out that LOCATE now supports automatic grouping of objects both in workstations and in elemental and fixed obstructions.

Other grouping issues needed to be addressed but time limitations prevented a more thorough treatment of them at this time. Manual grouping and ungrouping must still be added to LOCATE. The need for manual grouping has been considerably reduced, however, by the automatic grouping facility that allows users to fix objects within workstations and obstructions and to move everything about as a unit without invoking a grouping command.

Manual grouping will be important in allowing users to customise objects for inclusion in a specialised tool library. Also, users on occasion may create grouped objects that are not intended to function as obstructions and that simply reside in a workstation or outside workstations in the workspace area. In most cases, however, users will want to take advantage of the automatic grouping facility, letting LOCATE handle the grouping and ungrouping for them. This should save users considerable time in creating their designs.

LOCATE's Verification Checks

Before passing on to the minor refinements, a few comments are in order about the various checks that LOCATE is now able to do as part of its normal processing. Currently, two types of checks are identified, those done prior to running cost functions, to make sure that all the necessary data are present, and, those that detect whether a user's action will violate the conditions of the LOCATE software.

- **Cost Function Checks:**

- at least one Domain Weight is non-zero;
- link function modality data are present in all workstations where there exist non-zero domain weights for those modalities;
- the "Commit to all entries" flag is selected for all Priority Weight data.

- **Constraint Checks:**

- the height and width of elemental and fixed obstructions and of generic objects, as well as the radii of workstations, are > 0 ;
- a user is attempting to drag an object outside of the workspace boundary.

[LOCATE posts a warning to the user that the object is about to be placed outside the workspace boundary and the user then may decide to continue with the placement or cancel it.]

- the radius of a workstation's enclosing boundary is greater than or equal to zero ($r \geq 0$).

[LOCATE displays an alert box prohibiting a setting where $r < 0$. That minimum value will to be changed from zero to 1% of the workspace.];

- a user is attempting to create or place an elemental obstruction outside of a workstation.

[LOCATE displays an alert and prohibits the creation, placement or re-positioning.]

- more than five elemental obstructions are created inside a workstation;

[LOCATE displays an alert and prohibits the creation or placement of more than five such obstructions.]

- a user is entering priority weight data for workstations for which there are no link function data.

[LOCATE permits the entries but posts a warning that no link function data have been entered for the workstation(s). If the user attempts to run a cost function without entering those data, the second cost function check, in the list of cost function checks, above, prevents the generation of a cost function until the user enters the data.]

Minor Refinements

The following minor refinements were identified for inclusion in this version of LOCATE:

- making cost functions, as displayed in the cost functions window, cumulative within and across sessions;
- expanding the "All Information Window" to include all instances of the type of object selected;
- adding "Reset" buttons to the Priority Weights and Link Functions Windows;
- cleaning up multiple screen refreshes;

- drawing elliptical obstructions as ellipses instead of as circles;
- adding the Length-to-Breath aspect for Fixed Obstructions;
- modifying the title of LOCATE's design window to match the name of the design;
- locking of buttons in the cost function window;

All of the above refinements have been made to LOCATE. The first item, making cost functions, as displayed in the cost functions window, cumulative within and across sessions, has been implemented but the display of those cost functions now appears in a different window, the Cost Function History Window. The introduction of that window was part of work done under a separate contract, which allowed for some development work in support of a usability study of the LOCATE application.

The Cost Function History Window not only displays a cumulative list of all cost functions run during a LOCATE session, but allows the user to select any of those cost functions and recall ("Open") its associated design.

In the example of the Cost Function History Window that appears in Figure 1, below, notice that the cost functions may be sorted in any one of three ways: ascending, descending and order of creation, that is, the order in which the cost functions were generated.

In addition to recalling designs associated with various cost functions, all cost function entries may be cleared at any point by the user. Doing so deletes all the cost function items in the window as well as the temporary files that store the associated designs.

Both the Object Info and the All Objects Info Windows were modified during this development phase. Now, any item selected in the workspace is displayed in the Object Info window along with its attributes and the objects it contains. Container objects include workstations and obstructions.

The All Objects Info Window now displays all objects of the type currently selected in the interface. Also included with all those objects are their attributes and any items they may contain. One exception to this last point that is true for both Object Info and All Object Info Windows is that if a workstation is selected, information will appear about its contents but not the contents of those contents.

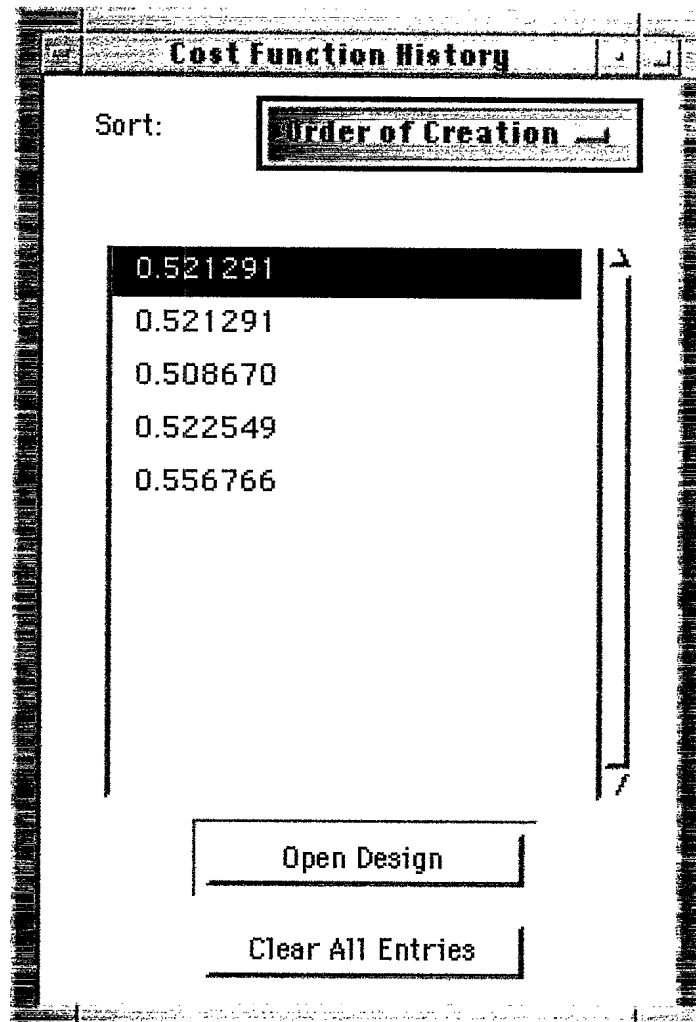


Figure 1. The Cost Function History Window

For instance, if a workstation contains an obstruction, information is displayed about the obstruction but not about the contents of that obstruction, such as the chairs and desks contained within it. To find out the contents of those items, the user selects the obstruction. To find out the details of objects within an obstruction, the user selects the individual items.

A user may now reset all values that have been entered into the Link Functions Window to their original default values by clicking on a "Reset" button in that window.

"Jumpy" displays, which occurred in several situations, and which were caused by unnecessary screen refreshes, have been fixed.

Elliptical obstructions, which had been drawn as circles in the interface are now drawn as they should be drawn, as ellipses. An added feature is the use of the Shift key to constrain both elliptical and rectilinear obstructions so that creating them conforms to the same rules for creating ellipses and rectangles using LOCATE's general drawing tools. That is, if the user holds down the Shift key while dragging out an elliptical or rectilinear obstruction, the obstruction is constrained to a circle or a square, respectively.

Some time was taken in addressing the next requirement, that of adding a Length-to-Breadth aspect for Fixed Obstructions. After a review of how fixed obstructions were handled in the original LOCATE code, it was determined that the standoff distance, for Type II obstructions, that is, fixed obstructions that have a Length-to-Breadth ratio much greater than one, is determined in a different manner than those of Type I that have a Length-to-Breadth ratio approximately equal to one. In the original code, it was necessary to separate the two types from one another on data entry.

Given the graphical nature of the current LOCATE interface, appropriate changes were made to the code to allow an interleaving of the two different types in the obstruction arrays. Also, different prompts to the user are necessary for the two types of obstructions and those different prompts now appear in the Fixed Obstruction Window, depending on the obstruction selected. Also, an addition to the Workspace Attributes Window now allows users to specify a threshold value for determining whether a fixed obstruction is a Type I or Type II obstruction.

Discussion with the Scientific Authority led to the possibility that the two types of fixed obstructions may be able to be combined into a single type. That possibility should be explored in future development work on LOCATE.

Saving or recalling designs now results in the name of the design being displayed as the title of the window.

Finally, buttons in the cost function window were locked so that if a user reduces the size of the window, the text of those buttons do not partially disappear.

Additional Refinements During the Contract Period

As is generally true in development, a variety of issues arise as work proceeds and one finds that requirements emerge that were only dimly perceived or anticipated. The automatic grouping, the layering of object types and the need for users to recall designs are a few examples. In addition, extensions to specified requirements, problems and bug fixes had to be addressed. Appendix A provides bulleted lists of other functional requirements that were addressed during this contract.

Summary

Major changes made to the LOCATE interface and to underlying LOCATE program during this work period include:

- implementation of an interface that allows LOCATE to export and import files to and from other CAD packages using the standard .dxf file format;
- saving of generic objects and their attributes;
- automatic grouping of objects within workstations and obstructions;
- a layering concept of different object types (workstations, obstructions and generic objects);
- a standard method of specifying relationships among objects within each of the three layers;
- standard editing functions of Cut, Copy Paste and Duplicate;
- addition of workspace boundary tools;
- multiple selection of objects;
- cost function and constraint verification checks
- minor refinements;
- additional refinements that emerged as the work proceeded (see Appendix B.)

LOCATE is now able to export and import files to and from CAD packages such as AutoCAD using the standard .dxf file format. Block definitions of LOCATE objects like workstations, S/R nodes, obstructions and generic objects allow users to create those objects directly in packages such as AutoCAD from a Block definition “library.”

Generic objects such as chairs, desks, cabinets and the like can now be created in LOCATE and saved with the design. Those objects may now be positioned, sized and rotated like other objects in the LOCATE interface.

Automatic grouping considerably reduces the need for users to use a group and ungroup function. Objects placed inside a workstation or an obstruction stay where they are placed and move and rotate whenever the workstation or obstruction is moved or rotated. The user naturally has the ability to select, move and rotate the object if it is not positioned as he or she would like it.

The automatic grouping is based on the concept that there are three types of object in LOCATE: workstations, obstructions and generic objects. The identification of those three types encouraged the development of the concept of layers within the LOCATE interface. The details of object layers are described in the section on refinements.

Specifying relationships among objects within layers is now done by the use of “Bring to Front”, “Bring Forward”, “Send to Back” and “Send Backward.”

A missing component in LOCATE prior to this phase of the work has been its support for the standard edit functions of Cut, Copy, Paste and Duplicate. Although LOCATE has provided support for cutting, copying and pasting text among the edit text boxes in various windows, it now supports those same edit functions for LOCATE’s three types of object: workstations; obstructions and generic objects.

Since those objects are complex entities which may contain other objects and a variety of function and argument data, the implementation of those edit functions was not a straightforward task. If a user copies and pastes a workstation which contains one or more obstructions, which themselves contain several generic objects, the resulting object pasted into the workspace will duplicate all link function data for the workstation, all function and argument data for all obstructions, all generic objects and will set links between the newly pasted workstation and all other workstations in the workspace. The only data not copied is priority weight data.

Only one group of LOCATE objects remained to be created at the beginning of this contract and that was the workspace boundary objects. Three workspace boundary objects (wall; window; and door) are now available to the user from the tool palette and function in the same way as other generic objects in the workspace. In adding those objects, the

workspace was expanded by an additional 10% to allow users room to place boundary objects inside or outside the boundary of the defined workspace.

Multiple selection of objects in the previous implementation of LOCATE was restricted to two objects only, which was necessary for the specification of priority weights between pairs of workstations. Selection has now been extended to multiple objects in the workspace.

Finally, several minor refinements were identified as requirements for this contract and all of those have been completed. A number of additional refinements emerged during the normal process of implementing the specified requirements and those were completed as well.

Next Steps

A variety of possible extensions to LOCATE's functionality emerged during this development phase. A prioritised list of those extensions has been compiled and is presented below. In addition to those items, recommendations for improvements and extensions also are available in the results of a usability study of the LOCATE interface, conducted under a separate contract (W7711-6-7320). Those recommendations and the items below provide ample data to inform decisions about the future development of LOCATE. Many of the suggestions here appear in the results from the usability study, but the two lists are by no means identical.

The items that follow have been arranged alphabetically by category so as to make review an easier task. Within those categories, an attempt has been made to list items in a kind of combined importance and ease-of-implementation order, which, of course, is suggestive rather than definitive.

Feature Extensions

Browser

- remove distortion between the Browser and the workspace area;
- provide a toggle to turn the Browser on and off.

Continuity with Older LOCATE Files

- ensure that example files from Hendy's Thesis and files from the Bridge Study work, which used an older version of LOCATE, are compatible with the current implementation of the interface.

Cost Function (Windows)

- make the default button in the Cost Function Window, “Cancel”;
- implement lists for incomplete user requirements in Cost Function Checks Window;
- investigate why changing the workspace dimensions and the Maximum Workspace dimension appears to have less impact on cost function values than one might expect.

Files

- collapse the current three-file format to a single proprietary file format;
- include textual descriptions in output files, e.g., object names, in a saved file;
- support exporting complex arrays from LOCATE into .dxf files and then importing them again after modification in other CAD packages;
- explore reading in new block definitions from .dxf files;
- have LOCATE detect and remove Ltemp files that may remain after a system crash.

Link Function Window

- develop a better display for data entry;
- change check boxes to radio buttons;
- save the state of the window and display it when the window is re-opened;
- add up-down buttons to allow users to display data for other workstations without closing and re-opening the Link Functions Window.

Links Display Window

- update different types of link that may be displayed and integrate that with the Links Display Window;

Menus

- update the procedure for determining the minimal boundary of workstations and other objects and re-enable the “Minimize Boundary” command in Execute Menu;
- explore the rationale for a global “Minimize Boundary”;
- determine if drop-down menus in Motif/PC windows should stay down when selected;
- put “Close” command back in the File Menu for the Macintosh version;
- disable menu items as appropriate.

Object Info Windows

- extend updating function for “All Objects Info” Windows;
- add a real-time display of x, y, & theta values. Display should be in the “Object Info” Window and/or in a status window attached to the main design window;
- consider making the “Object Info” Window an editable window;
- investigate further the usefulness of the Object Info and “All Object Info” Windows;
- determine why scrolling starts off slow in the All Objects Info Window and then speeds up.

Object Manipulation

- integrate a standard grouping function with LOCATE’s automatic grouping;
- eliminate expanded handles on rotated objects;
- implement Ctrl-clicking a selected palette tool to bring up the palette library window. Currently, if the tool is already selected, another tool must be selected first and then a Ctrl-click performed back on the target tool;

- explore how to move obstructions, and their contents, in and out of workstations (see “Obstructions”, below). Determine if the four obstruction types, which are part of the current implementation, can be consolidated.

Obstructions

- investigate how best to modify the display of obstructions: should there be a “snap-to” option? Should there be a border of a certain pixel width?;
- investigate how obstruction types might be collapsed and, consequently, best represented in the palette.

Other (Generic) Objects

- make the height of the desk object slightly higher than its current default, so as to accommodate comfortably other objects like a computer;
- implement better display representations for generic objects in the palette.

Palette

- implement at least one user-defined palette for a different problem domain;
- add a rotation tool in the palette;
- explore how users might add customised objects and icons to the palette.

Priority Weights Window

- display workstation names in the Priority Weights Window instead of “1st workstation” and “2nd workstation.”;
- add up-down buttons that allow a user to display data for other workstation combinations without closing and re-opening the Priority Weights Window.

Source/Receiver (S/R) Node

- correct problem in which shrinking a workstation leaves an S/R node outside the workstation boundary;
- consider adding an attributes window for S/R nodes.

Windows (General)

- change displays of numbers to decimals instead of scientific notation;
- display only three decimal values for numbers in dialogue boxes;
- incorporate defaults for functions and arguments and place them in the Link Functions, Priority Weights, Elemental Obstruction and Fixed Obstruction Windows;
- opening a previously saved design should restore the workspace view that was in effect when the design was last saved;
- when double-clicking on numbers in edit text boxes select the entire number, not just up to the decimal;
- add “Apply” buttons to windows;
- add up-down buttons in workstation and obstruction attributes windows to allow users to navigate to different workstations and obstructions; do the same for Link Functions and Priority Weights Windows;
- left justify items in edit text boxes;
- implement an autogrid feature in which objects automatically align to LOCATE’s grid;
- review all windows for appropriate buttons, close boxes, etc.;
- determine and implement appropriate system responses for link colours when a user turns off all Domain Weights;
- provide users with better support for the data entry process.

Workspace Attributes Window

- do not allow the name of the design to be editable;
- modify the Domain Weights section so that the system will check entries before closing the window to determine whether “1’s” or values summing to “1” have been entered. For example, determine how the system will respond if, say, the following two values are entered: .6 and .4 for Auditory and Visual modes, respectively, but only the Visual check box is checked.
- determine and implement appropriate system responses for link colours when a user turns off all Domain Weights;

Workspace (Design) Window

- alter link colours when a new domain weight is added;
- implement a better font for menus, etc.

[Note: The old default font was changed in the most recent version of Neuron Data’s Elements Environment.]

- provide support for multiple design windows;
- when a user switches from one workstation or obstruction to another in the object’s attributes window, automatically centre the object in the display window and select it;
- add a real-time display of x, y, & theta values. Display should be in the “Object Info” Window and/or in a status window attached to the main design window;

Workstations

- fix a problem in which the resizing down of a workstation leaves its S/R node outside the workstation boundary;
- set the minimum value for the enclosing boundary of workstations to 1% of the size of the workspace.

Miscellaneous

- review and determine the appropriateness of the coordinate systems for all LOCATE object types;
- solve problem related to changing zero-point in the middle of creating a design.

System Checks

In addition to the variety of checks that were incorporated into LOCATE during this work phase, several other checks need to be implemented, namely checks to determine whether users:

- are trying to place objects outside the workspace boundary. If a user attempts to do that, using the object's attributes window, LOCATE should post a warning to the user but should allow the placement if the user decides that is where the object should be located. Current support provides for such a warning only if the user attempts to drag an object outside the workspace boundary;
- are trying to position S/R nodes or Elemental Obstructions outside of a workstation. This should be prohibited. Current support disallows this if a user attempts to drag and S/R node or an obstruction outside. Users can only drag those objects to a point where their centre points are at the edge of the workstation boundary;
- have exceeded the bounds of the Maximum Workspace Dimension. For example, can a user override the Maximum Workspace Dimension and enter a value that is smaller than the computed value for the given dimensions would be? How small can that value be? Can it be 0?

A Common Interface

The LOCATE application is one element in a plan for a broad program that will provide software support for military missions. The program will support such things as mission analysis, scenario generation, function analysis, function allocation, task analysis, network analysis, etc.. In that context, LOCATE would be used by engineers to design workstation layouts and to help them evaluate the effectiveness of their designs.

LOCATE is now at a stage where broader program issues can be addressed. A key concern in that program will be the consistency of the interface among its software components. The Motif look is already a standard but increasing consideration should be given to common interface choices within that standard.

For LOCATE, that should include the development of its basic functionality as well as its on-line, intelligent help system. All aspects of the software should be examined with achieving that commonality, where possible and desirable. During the course of the current contract common interface issues were discussed for the following LOCATE components: 1) the palette; 2) editable and non-editable fields; and, status fields.

Other Development Work

In addition to the work completed under the current contract, other development work has occurred during this contract period. In a usability study of the LOCATE application (W7711-6-7320), an important extension was added that had not been anticipated at the beginning of that contract. That was the addition of a cost function history which allows users to recall previous designs from the cost function values generated from those designs. A Cost Function History Window now lists those cost function values and users may select any one of those values and recall its associated designs.

In another study conducted during this period, the fundamentals of a hypertext help system were added to LOCATE. LOCATE help files were placed at an Internet Website and can be accessed directly through a menu item in the LOCATE application. Selecting "LOCATE Help" invokes a customised Web Browser that logs a user on to the Internet and connects to the online help site for LOCATE. The Web Browser may also be used to access the same help files locally.

Another addition from that study was the integration of the LOCATE interface with Neuron Data's Intelligent Rules Element. That connection has established a link between LOCATE and an underlying rule and object based expert system. The study demonstrated a proof of concept that LOCATE's help system can make use of a reasoning component to provide intelligent aiding to LOCATE users.

In conclusion, the development of LOCATE has proceeded along three lines simultaneously during this contract period. The current contract added a number of important features such as an interface to other CAD packages, multiple selection of objects, an automatic grouping facility, the concept of layering and standard edit functions of Cut, Copy, Paste and Duplicate. Major contributions from other related work included the ability to recall previous designs and an intelligent help component including direct access to online, Internet help.

Two of the contract efforts have provided recommendations for possible future development work on LOCATE. Now that most of its basic functionality is in place, efforts can be directed to developing a mature product and to integrating LOCATE with other tools into a broader program for effective support for military missions.

References

- Hendy, K. C. (1984). 'LOCATE': A program for computer-aided workspace layout. Master's Thesis, Department of Electrical Engineering, Monash University, Clayton, Victoria, Australia.
- Hendy, K. C. (1989). A Model for Human-Machine-Human Interaction in Workspace Layout Problems. *Human Factors*, 31(5), 593-610.
- Hendy, K. C., Berger, J., & Wong (1989). Analysis of DDH280 bridge activity using a computer-aided workspace layout program (LOCATE). DCIEM Report # 89-RR-18. Department of National Defence. Defence and Civil Institute of Environmental Medicine, Downsview, Ontario, Canada.

Appendix A
Annotated List of
Source Code Files for
the LOCATE Interface:
Macintosh Implementation



Artificial Intelligence
Management and Development Corporation

Notes:

- i) .c files are C source code
 .obj files are compiled object code
 .rc files are Open Interface resource description files
- ii) Some extraneous files in the Locate folders are outdated and will be erased in future.

ALLOBJIN.C ALLOBJIN.RC

- The “AllObjInfo” module contains the code and Object Info resources necessary for the “All Objects Info” window.

AOBS.C,, AOBS.RC

- The “AObs” module contains the code and Object Info resources necessary for the “Fixed Obstruction” window.

ASSIGN.C,

- Original Locate C file

CFALERT.C, CFALERT.RC

- The “CFAlert” module contains the code and Object Info resources necessary for the alert box that appears when there have been changes to the design but no cost function has been run since those changes have been made. The alert box appears before displaying the Cost Function History window .

CFBROWSE.C,, CFBROWSE.RC

- The “CFBrowse” module contains the code and Object Info resources necessary for the “Cost Function History” window.

CFCHECK.C, CFCHECK.RC

- The “CFCheck” module contains the code and Object Info resources necessary for the “Cost Function Checks” window.

COSTFN.C, COSTFN.RC

- The “CostFn” module contains the code and Object Info resources necessary for the “Cost Function” window.

DRAWROTD.C, *DRAWROTD.H*

- C code for handling the drawing of rotated objects

DXF.H

- Header file with DXF format constants

EDITOR.C, EDITOR2.C, EDITOR.RC

- The “Editor” module contains the code and Object Info resources necessary for the main Locate window (includes code for Diagrammer, palette, rulers).

EVAL1.C

- Original Locate C file

EWATTR.C, EWATTR.RC

- The “EWAttr” module contains the code and Object Info resources necessary for the “Elemental Workstation Attributes” window.

EXTERN.H

- Original Locate header file

FORMAT.H

- Original Locate header file

FUNCT1.C

- Original Locate C file

HEADER.DXF

- Contains information that gets added to all exported DXF files

IMPRTDXF.C

- C code for handling the importing of a workspace from DXF format

INFOWIN.C

- C code for handling the “Object Info” window

INTFC.H

- Header file with prototypes for interface functions

LINKDISP.C, LINKDISP.RC

- The “LinkDisp” module contains the code and Object Info resources necessary for the “Link Display” window.

LINKFNS.C, LINKFNS.RC

- The “LinkFns” module contains the code and Object Info resources necessary for the “Link Functions” window.

LOCATE.C

- Based on the original LOCATE.C file, this contains the code necessary for loading in a workspace and for computing the cost function.

LOCATE.DAT

- Object Info compiled resources that are used by the Locate application at run-time.

LOCATE.EXE

- The Locate application

LOCATE.H

- Original Locate header file

LOCATE.π

- Locate project for Think C, version 7

LOCATE.RC

- Object Info resources in text format

LOCNEW.C

- C code for handling the creation of a new workspace

LOCNEWEW.C

- C code for handling the creation and deletion of workstations and obstructions

LOCSAVE.C

- C code for handling the saving of a workspace

LOCSVDXF.C

- C code for handling the saving of a workspace in DXF format

MAIN.C, MAIN.RC

- The “Main” module contains the “main” function which starts up the application.

MISC.C

- Original Locate C file

MISCRSRC.RC

- The “MiscRsrc” module contains Object Info resources needed by the application (primarily menu and icon resources).

OPTIM.C

- Original Locate C file

ORIGIN.C

- Original Locate C file

OTHEROBJ.C, OTHEROBJ.RC

- The “OtherObj” module contains the code and Object Info resources necessary for the “Other Object” window.

OUTPUT.C

- Original Locate C file

PAEDIT.C, PAEDIT.RC

- The “PalEdit” module contains the code and Object Info resources necessary for the “Palette Editor” window.

PRINTPRE.C, PRINTPRE.RC

- The “PrintPrev” module contains the code and Object Info resources necessary for the “Print Preview” window.

PRIORWGT.C, PRIORWGT.RC

- The “PriorWgts” module contains the code and Object Info resources necessary for the “Priority Weights” window.

PROJECTS.C, PROJECTS.RC

- The “Projects” module contains the code and Object Info resources necessary for the Projects & Groups window (currently disabled).

RULER.C, RULER.RC

- The “Ruler” module contains the code and Object Info resources necessary for the “Ruler” window.

SPLASH.C, SPLASH.RC

- The “Splash” module contains the code and Object Info resources necessary for the startup screen.

START.C, START.RC

- The “Start” module contains the code and Object Info resources necessary for the usability “Start” window.

WOBS.C, WOBS.RC

- The “WObs” module contains the code and Object Info resources necessary for the “Elemental Obstruction” window.

WSATTR.C, WSA.RC

- The “WSAttr” module contains the code and Object Info resources necessary for the “Workspace Attributes” window.

Appendix B

**Additional
Functional Requirements,
Problems and Bug Fixes**



Artificial Intelligence
Management and Development Corporation

The following categorised list identify requirements that emerged as development proceeded during this contract. Also included in those lists are problems and bug fixes that arose as a consequence of making other changes to the LOCATE application.

A few of the items below are repeated more than once, as they are pertinent to more than one category.

Additional Functional Requirements

Cost Function

- checks done prior to running a cost function:
 - check that all “commit all entries” is selected in all Priority Weights Windows;
 - checks that Link Function data are present for all non-zero priority weights;
 - checks that at least one domain weight is non-zero.
- Cost Function Window output default is cost function only; details are optional;
- add keyboard shortcut (Ctrl-F) to Cost Function Menu Item in Execute Menu;
- add “Cancel” button to Cost Function Window.

Link Functions

- Link Functions and Priority Weights Windows can be invoked by selecting either an S/R node or its surrounding workstation.

Menu Items

- temporarily remove “Minimize Workstation” from the Execute Menu.

Object Info and All Objects Info Windows

- make the “Object Info” and “All Objects Info” Windows modeless;
- check that “All Objects Info” Window does an Update when it is opened;
- display the names of objects in the “Object Info” and “All Objects Info” Windows;
- list S/R & Elemental Obstructions under “Contents” in Info windows.

Object Manipulation

- all objects are now scaleable by either directly clicking and dragging or by changing attribute values in dialogue windows;
- user may now select handles of objects to rotate them;
- user may now rotate any object; text is currently the exception;
- double-Click on an object to bring up its attributes window (from View Menu);
- after selecting a palette tool for an object, allow a click in the workspace to create an object of default size; the object may also be created by clicking, holding and dragging;
- moving anything in an obstruction should move the entire obstruction.
To move generic objects within an obstruction or outside it, the Ctrl key must be held down.

Obstructions (Elemental)

- make S/R node a configurable part of an obstruction;
- in the Elemental Obstructions Window, put Obstruction Name and Number first followed by the Workstation Name and Number;
- list S/R & Elemental Obstructions under “Contents” in Info windows.

Obstructions (Fixed)

- rearrange the Fixed Obstruction Window so that “r:” is changed to “Radius (Enclosing Boundary)” and put that where the L:B ratio buttons are now located. Those buttons can then be put under Width and Height and in a straight line rather than one underneath the other. That way, the window is expanded to the left and not downward, where we can’t afford the room.
- add automatic computation of the Radius of enclosing boundary in the Fixed Obstruction Window—update in a similar way to the Maximum Workspace Dimension in the WS Attributes Window; the field should be made non-editable.

Palette

- a text box now appears at the top of the palette giving displaying name of the currently selected tool;
- after selecting a palette tool for an object, allow a click in the workspace to create an object of default size; the object may also be created by clicking, holding and dragging;
- change the icon for the column tool in the palette.

Priority Weights

- Priority Weights and Link Functions Windows can be invoked by selecting either an S/R node or its surrounding workstation;
- choosing “Priority Weights” from the Links Menu when only one workstation is selected brings up a Priority Weights Window for specifying 1st-order links.

S/R Nodes

- make S/R node a configurable part of an obstruction;
- the Link Functions and Priority Weight Windows can be invoked by selecting either an S/R node or its surrounding workstation;
- list S/R & Elemental Obstructions under “Contents” in Info windows.

Text and Button Changes

- change “Import” and “Export” to “Import DXF” and “Export DXF”;
- change “Length” & “Breadth” to “Height” & “Width” in Fixed Obstruction Windows;
- Add Menu items: “Bring to Front” & “Send to Back” (“Bring Forward” and “Send Backward” are present);
- change names of “Info” and “All Objects” Windows to “Object Info” and “All Objects Info”;

- change constraint text in Workspace Attributes Window to:
 - “Prevent overlap”
 - “Between Workstations”
 - “Between Workstations and Other Elements (W:H~1)”;
 - “Between Workstations and Other Elements (W:H>>1)”;
 - “Between Workspace Elements and Straight, N-R Boundaries”;
- change the “Minimum” and “Maximum” default values from “900” to “10”;
- change the positions of the first three items in the Links Menu to read:
 - Link Functions
 - Priority Weights
 - Domain Weights
- clean-up wording in windows: remove “Centre Points of Obstruction”; change “Theta” to “Angle”, etc.;
- add title to window showing CF checks: “Cost Function Checks”;
- add “Run” button in the Cost Function Checks Window;
- add a “Print” button to the Print Preview Window;
- change title of Window “Workstation Obstructions & Functions” to “Elemental Obstruction and Attributes” in title of EOb window;
- change instances of “WOb” to “EOb”, e.g., in the Object Info Window reporting of the contents of a workstation;
- change the word “Ruler...” in the Align Menu to “Rulers...”;
- add the following to the bottom of the Splash screen; use a smaller font, as below:
 - © 1997 Her Majesty the Queen in right of Canada, as represented by the Minister of National Defence.

Workspace Attributes

- change positions of “Top” and “Bottom” in Workspace Attributes Window;
- compute the Maximum Workspace Dimension from the Left, Right, Top and Bottom values entered by a user;
- allow the user to override the maximum workspace dimension; revert to a computed value if the user changes and/or tabs out of one of the dimension edit text boxes;
- make all system countable items in the Workspace Attributes Window non-editable, that is, all the information in the bottom, left of that window.

Miscellaneous

- two redundant “link” tools removed from palette;
- temporarily remove “Open Projects” from File Menu;
- add a “Comments” attribute in attributes windows of selected objects;
- make “OK” button default in all windows;
- made Esc key default for “Cancel” button in all windows.

Problems Solved and Bugs Fixed (26)

Crashes

- problem of crash when clicking on elemental obstruction tools, and a shifting of tools in the Palette Library when Control Clicking the remaining tools; problem has arisen from removal of two link tools;
- moving S/R nodes inside an obstruction crashes the system;
- crash when a workstation is selected and Priority Weights are chosen from the Links Menu;
- dragging the mouse across the WB tools crashes the system.

Menu Items

- selecting “Minimize Workstation” in the Execute Menu, after clicking on a workstation, makes workstations inaccessible.

Object Info and All Objects Info Windows

- disappearance of “Object Info” Window.

Object Manipulation

- reloading of a saved file converting a rectilinear obstruction into a circle;
- unwanted shifting of objects inside workstations;
- objects not rotating as appropriate when their obstruction container is rotated;
- although dynamic rotation of an elemental obstruction rotates its associated S/R node, rotation using the obstruction’s dialogue box does not;
- unwanted changes that happen in the sizes and positions of “Other objects”;
- objects dragged off the screen can not be brought back into the display window.

Obstructions (Elemental)

- elemental obstructions disappear when moving around in an EW.

Obstructions (Fixed)

- although the width and height of Fixed Obstructions are visible in the “Object Info” Window, when Fixed Obstruction objects are selected those dimensions are not shown as editable attributes in the Fixed Obstruction Window, that is, nothing currently appears in the “Length” and “Breath” text edit boxes;
- Fixed Obstruction numbers appearing as negative values.

Printing

- problem with spooling multiple pages to disk when Print command selected.

Workstations

- dragging out a workstation appears to produce jumps in the size of the EW as it is being created;
- workstations suddenly inaccessible after clicking on a workstation and selecting “Minimize Workstation” from the Execute Menu.

Miscellaneous

- fix bug that sometimes puts entire path name into the filename edit text boxes in the Standard Save dialogue box and the WS Attributes window.
- problem of vertical lines appearing on Object Info Window;
- fix frozen scroll bars;
- option-click in the workspace no longer brings up old Diagrammer Menus;
- clicking minimise button in Motif/PC “look” no longer causes the design window to disappear;
- cursor “sticks” to a selected object;
- when saving and reopening files, the workstations do not appear; radius of enclosing boundary has been saved as “0”;
- “Maximum Workspace Dimension” has a different value than it should when reloaded.

#504207