Man-Computer Graphics in Preliminary Ship Design

Robert S. Johnson,* Arthur L. Fuller,† James A. Claffey,‡ Fritz R. Bjorklund§
Naval Ship Engineering Center, Hyattsville, Md.
and
Bernard M. Thompson¶
Naval Ship Research and Development Center, Carderock, Md.

The Naval Ship Engineering Center through its Computer Aided Ship Design and Construction Office is working toward maximum effective use of computers in the Navy's ship design process. Many batch type analyses were developed but certain naval architectural problems, such as ship hull form creation and arrangements, cannot readily be handled by the computer without human intervention. Hence, interactive graphics developments were required. The existing interactive graphics programing languages were applicable only to the hardware for which they were written. To minimize conversion problems of inevitable hardware configuration changes, a portable interactive graphics language was developed and used. The active graphics programs described herein are under continuing development and in use for hull form generation, deck and bulkhead location, and topside and internal room arrangements. In their present form, they are applicable to early stage feasibility and preliminary design. The Integrated Ship Design System (ISDS) is being developed to allow communication of all types of design programs, including interactive graphics through a common data base; so that all applicable preliminary ship design tasks can be done within the computer environment.

Introduction

IN 1967 the interactive graphics terminal for digital computers had become a fairly common device in universities and research laboratories. Herzog[1] at the University of Michigan and Sutherland[2] at MIT had begun to teach courses in how these terminals should be programed. Coons[3] at MIT had developed a surface representation technique clearly aimed at using an interactive graphics terminal as the input/output device. IBM had developed a language—GSP[4]—which could be used to program their 2250 series of interactive graphics terminals. The General Motors Research Laboratories had applied a Coons-like technique to the definition of automobile bodies using an interactive graphics system. Lockheed had defined some parts for the C-5A using an interactive graphics scope and had directly generated the numerical control tape to cut them. By 1969, McDonnell Douglas Automation Co. had implemented a system where they did their two dimensional plus depth parts programing right at the scope as their normal procedure, i.e., at the scope their parts programer generated an APT program directly from a geometric definition of the part on the scope.

Received June 13, 1972; presented as Paper 72-587 at the AIAA SNAME/USN Advanced Marine Vehicles Meeting, Annapolis, Md., July 17-19, 1972; revision received October 2, 1972. This paper summarizes work done by more than a score of engineers and programmers at the Naval Ship Engineering Center (NAVSEC) and the Naval Ship Research and Development Center (NSRDC). R. Chen of NSRDC developed the program referred to as "a parallel effort."

Index categories: Marine Vessel Design (Including Loads); Computer Technology and Computer Simulation Techniques.

*Assistant Technical Director and Head, Advanced Technology Branch, Ship System Design and Development Division.

†Supervisory Naval Architect for Computer Applications and Computer Aided Ship Design Coordinator, Advanced Technology Branch.

‡Naval Architect, Marine Engineer, Computer Aided Ship Design and Construction Office.

§Naval Architect, Computer Applications Section, Advanced Technology Branch.

¶Naval Architect, Computer Aided Ship Design Division, Computation and Mathematics Department.

By this time NAVSEC had approximately 200 FORTRAN computer programs which were suitable for use in preliminary and Contract Design of naval ships. However, little calendar time reduction in the design process was noted despite the large amount of automation of which had previously been manual calculation procedures. Johnson[5] discussed this in 1969 and identified why little time reduction had taken place. Principally, it was due to 1) the existing programs were not coordinated with one another; 2) information exchange between and among programs was still done manually; 3) certain design procedures were not amenable to automation using traditional batch programing methods, e.g., developing general arrangements or combat system block diagrams; and 4) data acquisition in preparation for processing had been ignored. In 1964, Coons[6] presented an abstract description of requirements for a fully automated computer-based design system utilizing Sketchpad[7] which included interactive graphics input/output.

Unfortunately, a piecemeal or incremental solution to the requirements implied by the four deficiencies listed is not available. It is only solvable by a massive effort. Hence, the Integrated Ship Design System (ISDS) was initiated under the Navy's Computer-Aided Ship Design and Construction (CASDAC) Program in 1969. A demonstration ISDS, which established the feasibility of the approach, was reported by Corin and Johnson[8] and documented in Ref. 9. The system part of the initial operational version of ISDS went up in May 1972. In England, I. M. Yuille[10] has initiated development of a similar system.

A principal subproblem of the ISDS effort was to develop a method for quick handling of those design procedures which were not amenable to traditional programing methods, such as 1) developing a combat system block diagram; 2) developing the heat-cycle diagram for a steam plant; 3) developing the ship's lines; and 4) laying out the general arrangements. Each of these procedures have four things in common: 1) they are complex in themselves; 2) they are intimately related to the over-all system, i.e., strongly coupled to other subsystems; 3) they can be highly creative efforts; and 4) their solution is depicted graphically.

Because of the first three attributes, it is extremely unlikely that either an algorithm or a heuristic procedure for their solution could be discovered and programed. Thus, some sort of approach that would utilize a high degree of rapid designer/machine interaction would have to be developed in order to automate them. Their last attribute required that the input/output medium incorporate a graphic display. Such a device is the programable interactive graphics computer terminal. A good over-all summary of different types of interactive graphics terminals has been presented by Lansburg and Seibold.[13]

While the solution of such problems must be depicted graphically, the results must be easily transferable to other parts of the design. For example, once the external geometry of a ship is defined, it is required by the designers of 1) the antenna, 2) the machinery, 3) the propellers, 4) the structures, and 5) the arrangements, etc. Solutions which do not facilitate this information exchange do little to speed up the over-all ship system design process. This is one reason why a system like ISDS was required.

In addition to the ability to describe something graphically and to have that graphic image processable by other engineering procedures, other requirements for graphics programs were recognized. Some of these requirements were known at the outset of the work, and others were developed during the work. They will be compiled here for convenience. 1) Because of the manpower investment necessary to develop the programs, they should be as machine independent as possible, i.e., be portable. 2) The general definition of the ship's geometry and other data attributes must be available at the outset before programs are developed. 3) Since some graphics coding becomes repetitive, a macrocommand language suitable for ship design is required. 4) Many types of hard copy are required at various intervals of the design process, which can range from a Polaroid snapshot to a contract drawing. The acceptable time and cost spent generating them covers a wide range. 5) Too much information on the screen at one time is confusing and distracting. 6) The response time that a user will tolerate is important in over-all graphics program design. 7) The display on the screen should be semitutorial. 8) The program should be foolproof. 9) A picture to be modified should be presented to the user, rather than a blank screen and a light pen.

Requirements 1–4, led to the development of the Interactive Terminal Interface System (ITIS), the Ship Geometry Package, and the Computer Graphics Ship Arrangements Program (COGAP) and several other interactive graphics systems and programs. ITIS, or ITIS.2 as the current version is called, is a software system for writing portable interactive graphics programs. The Ship Geometry Package provides for interactive hull geometry and major subdivision definition. COGAP is an interactive graphics program for detail ship arrangements.

Requirements 5–9 imply that the programs should be human engineered. This is perhaps the greatest lesson learned. If the programs are to be used by the ordinary, noncomputer-oriented, working engineer, then he must not be frustrated by them. If he becomes frustrated, he will simply do the job another way and the best programs in the world would perhaps go unused.

First, the ITIS and the concept of portability will be discussed. Then, the Ship Geometry Package and COGAP—which are being written in ITIS, will be described. Finally, their integration into ISDS will be discussed briefly.

Portable Supporting Software Development

A portable computer program or system software package can be defined as one that can be transported from the computer installation on which it was developed to another one with only minor changes. The principal purpose of this, of course, is to ensure that once the software is developed, it will not be rendered useless by a change in hardware configuration or type. This prolongs its useful life and expands its user community.

There are two major requirements for the development of portable computer programs: First, a strong commitment to the value of portable programs to overcome what initially appears to be disadvantages (additional programing and less efficient hardware utilization); second, a programing language which is generally available on various computer configurations. The programing language for nongraphic work which best fits the second requirement is FORTRAN. Within a FORTRAN program, if certain restrictions are observed and if input/output devices are handled carefully, its portability will be considerably increased. Therefore, there has been an increasing trend toward requiring programs to be written in FORTRAN.

The initial interactive graphics programing languages were applicable only to the hardware for which they were written. The use of these programing languages for interactive graphics would even make application programs written in FORTRAN hardware dependent. In order to address this problem, the Computer Aided Ship Design and Construction Office of NAVSEC and the Lockheed Georgia Systems Sciences Laboratory developed the ITIS.[11] The major function of ITIS is to provide a stable interface for graphics application programs on all major computer hardware configurations. Requirements were later added to address the complex data handling needs of interactive graphics.

The interactive graphics language has two major functions to perform: 1) it must allow description and control of a displayed image, and 2) it must allow the user to interact with the computer. Typically, this is done from a FORTRAN program via calls to subroutines.

ITIS FORTRAN calls can be divided into four major groups to illustrate the functions required. The first group serves as graphics initialization and graphics termination:

LOGON (LOG, N2IMG, N3IMG). Called once per run. First of ITIS routines to be called, performs initialization on LOG, an array used for communications between ITIS subroutines.

UNLOG (LOG). Called at end of ITIS job. Will clear CRT and release console.

VIEW2 (LOG, IMAGE, ID, LNG), initializes IMAGE, an array used to store information about a picture.

CHVU2 (LOG, IMAGE, IWND, ISIS, IUSER ITYP, MODE), changes window size, user coordinate system, and performs scissoring.

The second group of routines can be used to draw images:

D2LNE (LOG, IMAGE, KEY, KT, IXO, IYO, IX, IY), draws straight lines between points specified in IXO, IYO, IX, IY, where these parameters may be arrays.

D2PNT (LOG, IMAGE, KEY, KT, IX, IY), draws point(s) at location (IX, IY).

D2SEG (LOG, IMAGE, KEY, KT, IX1, IY1, IX2, IY2), draws line segment(s) with origins at (IX1, IY1) and end(s) at (IX2, IY2).

D2MOV (LOG, IMAGE, KEY, IX, IY), keyed item is moved to coordinates (IX, IY).

DALPH (LOG, IMAGE, KEY, KT, IX, IY, IALF), displays alphanumeric data.

DNUMB (LOG, IMAGE, KEY, KT, IX, IY, NUMB, ITYP), displays numeric data.

The third group of routines helps manipulate images:

SHOW (LOG, IMAGE, KEY), displays all items in an image which are keyed for status.

VEIL (LOG, IMAGE, KEY), disables light pen sensitivity of item(s) specified.

HIDE (LOG, IMAGE, KEY), turns off display of image or item.

REMOV (LOG, IMAGE, KEY), deletes image or item.

The last group of routines allows user interaction:

GPICK (LOG, ID, KEY, KT, KYLST, MODE), allows a light pen pick and returns with the KEY and ID of the item picked.

GALPH (LOG, IMAGE, KEY, KT, IX, IY, IALF, MODE), allows alphanumeric input to the program from a keyboard.

GNUMB (LOG, IMAGE, KEY, KT, IX, IY, NUMB, ITYPE, MODE), allows numeric (real or integer) input to the program from a keyboard.

G2COR (LOG, IMAGE, IX, IY, MODE), provides a tracking cross capability.

The subroutines just outlined provide the tools for the programer to display an image, to change it, and to allow interaction with the user. These are functions common to most graphics systems. In order to assure portability, some graphics features which are available on some machines and not others are excluded.

The 3-dimensional (3D) interactive graphics capability was implemented by doing a small amount of vector analysis, and then calling the applicable ITIS.2 2D routines. The 3D subroutine calls are essentially the same as those in 2D with an additional coordinate. Routines are provided to display any one or the three orthogonal views simultaneously from a selected viewing angle. In addition, a 3D tracking cross is implemented.

One object of ITIS was to design a system that was human engineered for both the programmer and the user. A definite effort was made to provide as many subroutines as was necessary to accomplish the required functions, but no more. In other words, if the programer has one logical function he wishes to perform, ITIS performs it with one subroutine call whenever possible. The example in Table 1 illustrates sequence required to enter a number from the keyboard in IBM 1130 GSP.

The end result of this human engineering effort is a language that is very easy to use, results in fewer errors during writing and use, and allows both programer and user to concentrate on the job at hand, not on the computer system.

Ship Geometry Package

The problem of defining the ship's geometry was broken into four parts: generation of lines to the weather deck, laying in the decks and main transverse bulkheads, definition of the superstructure and location of weapons, and laying in of individual compartments. A movie[14] showing the implementation of this package on an IBM 1130/2250 system has been produced. The programs are currently being implemented on a CDC 1700/274 which is operating as an on-line peripheral to a CDC 6700. In this new implementation, important but nondramatic improvements are also being made to the package.

Table 1 Programing comparison

```
    CALL SATNS (ATTNSOURCE)
    CALL DFMSG (CORVAL, COUNT, SIZE, INITVAL)
    CALL ICURS (CORVAL, CHARPOS)
    CALL RØATN (IATN)
90 CONTINUE
   IF (LATN(2)-(3)) 100, 200, 100
100 PAUSE
   GO TO 90
200 CALL RCURS
    CALL TLMSG (CORVAL, TEXT, ELCOUNT, TEXTCODE)
    CALL BCNV (INPUT, OUTPUT, CNVCODE, TEXTCODE)
whereas in ITIS,2, the call is
    CALL GNUMB (LOG, IMAGE, KEY, KT, IX, IY, NUMB,
      ITYPE, MODE)
```

If you wish to display the number after input, the ITIS,2 call is the same, but GSP requires at least one more call.

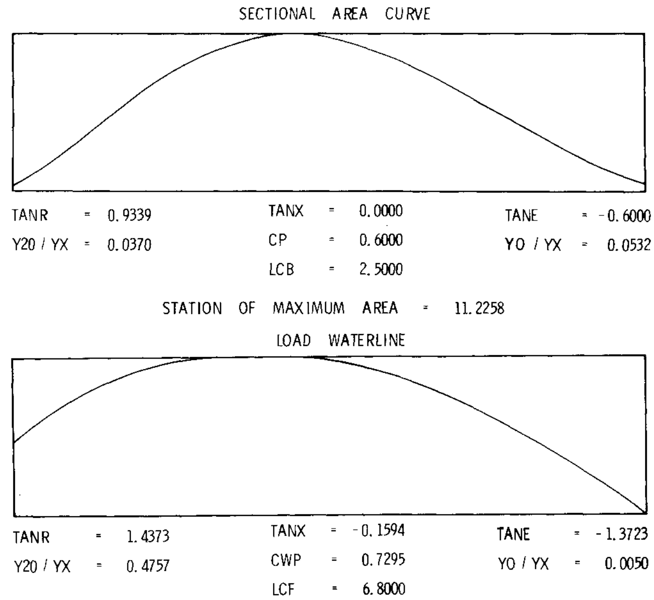

Fig. 1 Sectional area and load waterline curves.

All of these programs are aimed at Preliminary Design as opposed to detailed design for construction. The outputs from this initial design process are, in fact, input to the next design step.

Lines Generation

As ship synthesis model computer programs[5] which generate feasibility level ship designs from inputs like speed, endurance and payload came into more and more use, a need for a first guess at the hull form for a particular design became necessary. This was because the synthesis models do not satisfy all constraints, e.g., topside arrangements, and sometimes it was necessary to quickly (say, three hours) validate whether or not the design was "feasible" with respect to these normally unchecked constraints.

The lines generation program allows the user control, interactively via the scope, over the shape and characteristics of the section area curve and the Design Waterline (DWL). These curves are initially estimated by empirical relationships developed from a systematic study of destroyer hull forms which were originally developed for seakeeping studies.

The section area and DWL curves are each represented by a seventh order polynomial with eight boundary conditions to control their shape (see Fig. 1). The user, via the scope, has control over the boundary conditions and thereby controls the shape and other characteristics. The boundary conditions are the characteristics the designer usually defines or are defined by default for him by the synthesis model.

Boundary conditions for a seventh order polynomial representation of the section area and DWL curves are given in Table 2.

The section curves are defined at each station (up to 40 allowed) by two functions which are matched for ordinate and slope at the DWL. The boundary conditions governing the shape of these curves were a result of the previously developed Section Area and DWL curves, as well as empirical relationships governing halfsiding, deadrise angles, and flare characteristics.

The new version of the program allows user interaction and control of all the necessary boundary conditions that control the shape of each section in the hull. The program

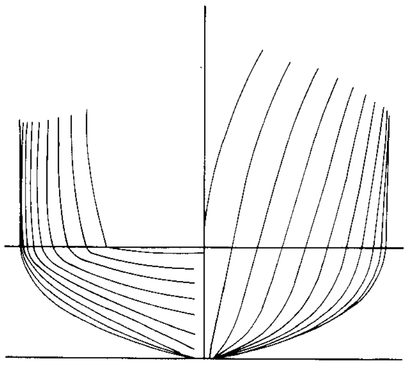

Fig. 2 Computer generated destroyer hull.

chooses initial values for the boundary conditions so that the user has something to start with. The user changes only the values that makes the desired changes to the hull. The resulting hull forms are fair since there are no discontinuities; i.e., even the keel rise curve is a curve with smooth transition from the keel line so that no longitudinal discontinuity in slope of the curve occurs. Hulls can be developed from basic hull form parameters in a matter of only a few minutes to perhaps an hour depending upon how many boundary conditions the user wishes to manipulate. Chines and similar discontinuities cannot be handled directly but can be reasonably approximated by fair lines (see Fig. 2).**

Decking Out

Given the hull form from the lines generation program and the location of decks, platforms, and major bulkheads, this program generates the offsets for the decks and platforms, computes the available arrangeable area for each major compartment, available tankage, and total hull volume. The major bulkhead input data are the machinery box and forepeak bulkheads. The program places a few other bulkheads as a result of its attempt to fit in decks and platforms while maintaining a specified bottom clearance.

The results are displayed on the scope as an inboard profile and table of areas and volumes. The user evaluates these results and alters the parameters in order to obtain the optimum feasible arrangeable area. This is a check of the synthesis model estimates for available areas.

Inboard Profile Topside Arrangement

This interactive program can be used to check the fore to aft topside arrangement of a prospective design study. Templates of various weapons and antennae can be manipulated on an inboard profile and plan view of the ship including superstructure, so that necessary clearances and interactions can be checked visually. As presently implemented, no analysis calculations on such things as weight, vertical center of gravity, or structural effects of the changes are being made.

Superstructure Shaping

Included in the Topside Arrangement program mentioned above is the facility to alter the shape of the superstructure, which is initially presented in a rectangular form, to give the necessary features desired, i.e., cutaways for gun clearances, boat wells, etc. The available arrangeable area in the superstructure can be checked against the required area. This is indicated by SS AREA REQ'D and OBT'D in Fig. 3.

A Parallel Effort

During 1969 and 1970, a parallel effort[12] was undertaken with the objective of defining the data exchange requirements of an interactive graphics package for defining the ship's geometry in the ISDS environment. The program could be used to perform arrangements of superstructure, weapons, antennas, and main transverse bulkheads. Most of the capabilities overlapped those of the aforementioned arrangements programs; however, different types of displays and programing methods were employed which provided a second perspective on the graphics task. The best techniques of both efforts have been used in subsequent graphics programs.

One unique capability of the parallel effort program was that it displayed floodable length curves over an inboard profile to assist in the location of main transverse bulkheads (see Fig. 4).

Computer Graphics Ship Arrangement Program

Having generated a geometric envelope and done some major subdivisions (lines generation and decking out), the next step in the design process is to do a more detailed arrangement study. COGAP is being developed to solve that problem. COGAP will not only allow for creation and arrangement of ship compartments, but will also be capable of creating and arranging component templates for the furnishings, machinery, electronic equipment, etc., that outfit the compartments. The program will be modular to permit future extensions into areas which interface with it such as routing of pipe, cable and ducting, and equipment and structural interference determination.

The program is initiated by inputting the three-dimensional deck, shell, and watertight bulkhead descriptions. This data, which would come from programs described previously, initiates the design data file into which are stored all data generated by COGAP during the design. The user inputs statistical data on cards and compartment and component template geometry data directly from the graphics terminal using the light pen and keyboard. Equipment templates are specified as combinations of five primitives; wedges, cylinders, cones, parallelepipeds, and spheres. Any combination of these primitives may be designated a template which may be manipulated as a single entity. The user can selectively retrieve (query) components by their statistical data at the scope.

Each template will be permitted two descriptions: approximate and detailed. The approximate description consisting of bounding cylinders and rectangular parallelepipeds will be useful for creating working displays for the graphics console; the use of approximate templates will minimize flicker, save buffer storage, and improve re-

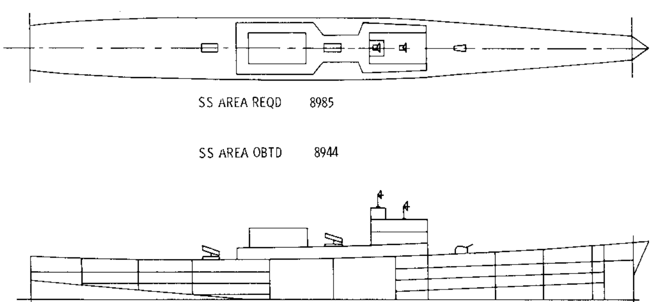

Fig. 3 Plan view and inboard profile.

**The figures in this report were retraced from actual computer plots in order to make them reproducible for printing.

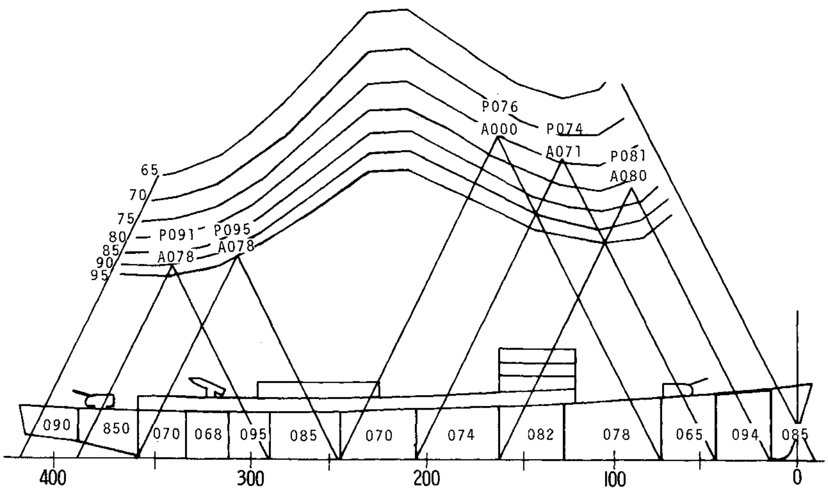

Fig. 4 Display for floodable length analysis.

sponse time. The detailed description will be used for accurate computations and hard copy plots.

Any template may be moved to any location, and rotated as desired in three dimensions. COGAP initially receives data describing the ship geometry hull, decks, and main transverse bulkheads—from punched cards. The user may specify additional decks and main transverse bulkheads, or he may delete or respecify existing decks and bulkheads. Thus, he may compartmentalize the ship with these surfaces and, with an additional operator-defined surface, the partition. Within a compartment, the operator may insert and position components. Thus the program can be used to arrange the compartments on a deck and completely outfit each compartment. A library of commonly used templates will be maintained.

Data on the arrangements, ship geometry, components and other features are kept on a mass storage device under the control of a COGAP subsystem—the Data Access Mechanism (DAM). DAM is designed to permit COGAP and other computer programs to store and retrieve information in the COGAP data base. Therefore, DAM is a mechanism whereby many ship design programs can share a common data base.

The use of hidden lines is important in problems involving three-dimensional objects, but the computation time and corresponding response time delay required to determine hidden lines can be objectionable to a user if the hidden lines are suppressed all the time, especially when many objects are being displayed. Suppression of hidden lines is not needed in many operations. COGAP is being implemented initially to display wire frame views without hidden line suppression. COGAP will permit the user to draw in three views simultaneously; front, side, top. The user is allowed to make changes in any view and to see the corresponding changes occur in the other views, which should help alleviate the orogram of visualization.

Table 2 Variable boundary conditions

Section area	Design water line
1. Cp-prismatic coefficient	Cx-maximum section coefficient
2. LCB—longitudinal center of buoyancy	LCF—longitudinal center of flotation
3. Bulb area at forward perpendicular	Halfbreadth at forward perpendicular
4. Station of maximum area	Halfbreadth at station of maximum area
5. Area at after perpendicular	Halfbreadth at after perpendicular
6. Entrance angle (slope)	Entrance angle (slope)
7. Run angle (slope)	Run angle (slope)
8. Slope at station of maximum area (0.0 by definition)	Slope at station of maximum area

Integrated Ship Design System (ISDS)

Design Environment

The objective of ISDS is to reduce the elapsed time of preliminary ship design by providing coordination among the application programs. Two complementary avenues have been pursued to accomplish this coordination: 1) A Ship Design File (SDF) serves as the central repository of all data related to the design, allowing automatic exchange of data among programs and affording a single source for up-to-date information on all aspects of the design. 2) Software support is provided, at a system level, to accomplish tasks which facilitate the coding and operation of the various programs.

ISDS is thus a software environment in which numerous application programs can be embedded. Some of these programs, described previously, involve interactive graphics while others are time-shared programs operated from teletype consoles. Figure 5 is a simplified version of one of many possible program call sequences which might be used at the start of the preliminary ship design process.

Data Exchange

The Ship Design File (SDF) is an ordered set of data which contains the physical description of the ship and its components, and the results of engineering analyses included in its design. At the outset of a design, the SDF is nearly empty, containing only a description of the mission requirements and the results of a ship synthesis model program. Each application program, as it is executed, draws input data from the SDF, complements this input with information supplied by the engineer, performs its design task, and returns its output to the SDF for use by subsequent programs. At the conclusion of the design process, the SDF contains a complete digital description of the preliminary ship design.

Many of the application programs, particularly the active graphics programs, require access to enormous quantities of data. It is imperative that this data be maintained in secondary storage under a data management system that can very rapidly retrieve into the main frame core those elements of data that are required by the program. Secondary storage becomes effectively a great extension of core storage. ISDS provides a random access data management system which provides such a capability for the graphics data as well as all the other data re-

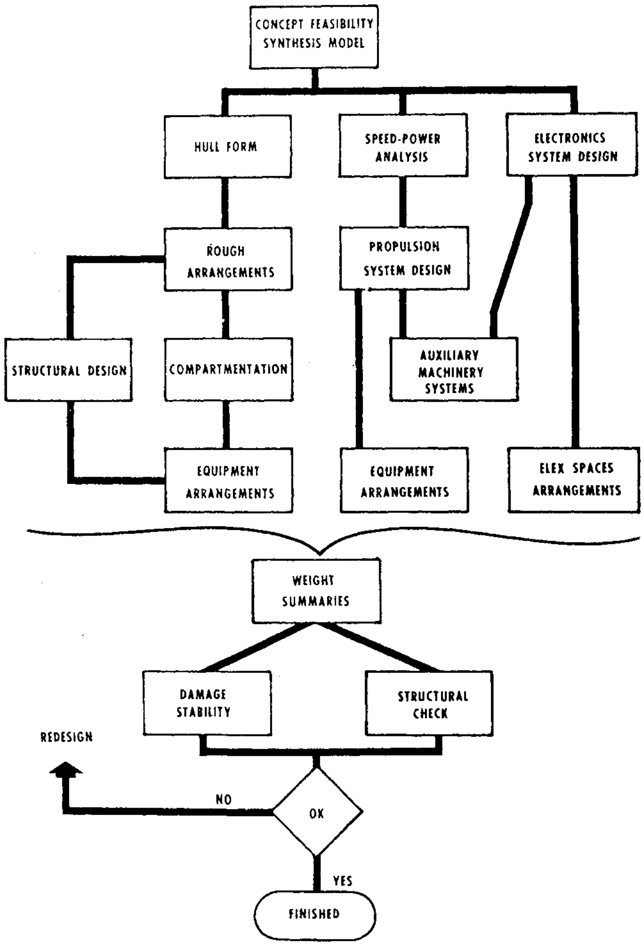

Fig. 5 One possible sequence of calls for ISDS.

quired to do the design. Information is organized into blocks of related data, e.g., data needed to define a compartment, a bulkhead, a machinery component, or a table of offsets. Blocks contain attribute data (e.g., weight and dimensions of a component) and pointers. Pointers identify the blocks of its bounding bulkheads; other pointers identify blocks for components located in the compartment. These pointer relationships are as important in building the digital ship model as the physical attributes.

Standard design information applicable to any ship design is stored on catalog files in a form usable directly by the appropriate programs. Catalogs exist for such data as hull resistance curves, properties of structural elements, and characteristics of off-the-shelf equipment items. A query capability is available to quickly identify equipment which will meet attribute specifications supplied by the engineer.

Software Support

The data management facility mentioned above is a most important aspect of the ISDS software support. Other areas are 1) design administration facility to provide file and user security and to record progress on the design; 2) automatic handling of the various data files used by any program; 3) capability of running several design cases in parallel, then picking the optimum for inclusion in the Ship Design File; 4) communications package to monitor time-sharing operation from a remote teletype console; 5) facility to allow user interface in a problem-oriented language.

Summary

We have defined and programed many of the early stage ship design tasks. In addition, much work has been done on computer systems software which now makes it possible to have another significant positive effect on the computer-aided ship design process. The task of putting these available resources together in order to improve the Navy's ship design capabilities is now underway.

References

[1]Herzog, B., Lecture Notes on Interactive Computer Graphics for Designers, Papers Number 1, 5, 6, 8, Course 6713, June 1967, Dept. of Naval Architecture and Marine Engineering Summer Conference, Univ. of Michigan, Ann Arbor, Mich.

[2]Sutherland, I. E., "Computer Displays," *Scientific American,* Vol. 222, No. 6, June 1970, pp. 57–81.

[3]Coons, S. A., "Surfaces for Computer Aided Design of Space Forms," Project MAC; Rept. MAC-TR-41, June 1967, MIT, Cambridge, Mass.

[4]"IBM 1130/2250 Graphics Subroutine Package," File 1130-25 Form C27-G934-1, Data Processing Div., IBM Corp., White Plains, N.Y.

[5]Johnson, R. S., "Automation in Pre-Contract Definition Ship Design," *Naval Engineers Journal,* June 1969, pp. 89–98.

[6]Coons, S. A., "An Outline of the Requirements for a Computer-Aided Design System," *SJCC Proceedings,* May 1963, pp. 299–304; also *SNAME Transactions,* Vol. 72, 1964, pp. 5–13.

[7]Sutherland, I. E., "Sketchpad, A Man-Machine Communication system," *SJCC Proceedings,* May 1963, pp. 329–346.

[8]Johnson, R. S. and Corin, T., "The Integrated Ship Design System, A Progress Report," *Technical Report of the American Ordnance Association,* April 1970, Sec. L, AOA, Washington, D.C.

[9]Corin, T. et al., "The Integrated Ship Design System Model 0 Documentation," Vols. 1–27, CASD Div., Computation and Mathematics Dept., Naval Ship Research and Development Center, Carderock, Md., Nov. 1970.

[10]Yuille, I. M., "A System for On-Line Computer Aided Design of Ships-Prototype System and Future Possibilities," *Proceedings of the Royal Institution of Naval Architects,* Paper 6, 1970, The Gresham Press, Old Woking Surrey, England.

[11]Fuller, A. L. and Milkowski, T., "The Interactive Terminal Interface System," *Technical Report of the American Ordnance Association,* Aprit 1970, Sec. Z, AOA, Washington, D.C.

[12]Thomson, B. M., Chen, R., and Skall, M., "Integrated Ship Design System Interactive Graphics," *Proceedings of the XXXVI SHARE Conference,* March 1971, pp. 289–301.

[13]Landsburg, A. C. and Seibold, F., "Interactive Computer Graphics in the Marine Industry," Presented to the Chesapeake Section of SNAME, April 1972, U.S. Dept. of Commerce, Maritime Administration, Office of Ship Construction, Washington, D.C.

[14]Ince, J., Fuller, A., and Johnson, R., *Interactive Computer Graphics in Ship Design* (Movie), NAVAIR Photographic Center 160SX, Nov. 1971, Naval Ship Engineering Center, Computer Application Section 6114E, Hyattsville, Md.