AIAA 95-3414
Results of a Flight Simulation Software Methods Survey

E. Bruce Jackson
NASA - Langley Research Center
Hampton, VA

A I A A Flight Simulation Technologies Conference

August 7 - 9, 1995 / Baltimore, MD

RESULTS OF A FLIGHT SIMULATION SOFTWARE METHODS SURVEY

E. Bruce Jackson*
NASA Langley Research Center
Hampton, Virginia 23681-0001

Abstract

A ten-page questionnaire was mailed to members of the AIAA Flight Simulation Technical Committee in the spring of 1994. The survey inquired about various aspects of developing and maintaining flight simulation software, as well as a few questions dealing with characterization of each facility. As of this report, 19 completed surveys (out of 74 sent out) have been received. This paper summarizes those responses.

Introduction

For many years, flight simulation modeling was an art practiced by a relatively few skilled engineers situated near large mainframe machines fast enough to perform real-time simulations. With the revolution in computer technology in recent years, the capability of solving rigid-body aircraft equations of motion with high-fidelity aerodynamic, propulsion, and control system models has arrived on many desktop computers. This increased computational capability is resulting in a proliferation of flight simulation environments and coding conventions.

Efforts have been made by the Flight Simulation Technical Committee to move toward setting standards for simulation models. Standard variable names, for example, would make it much easier to share simulation models between different simulation sites; it is also hoped that standards for modeling would increase the reutilization of simulation software, leading to improved productivity. Aerospace academics would benefit as well, if flight simulation software could be readily shared between industry and universities for teaching purposes.

As an example of the need for standards, the High-Speed Research program, a joint NASA-industry effort to develop a next generation supersonic transport, has an unprecedented number of participants, each with real-time and batch flight simulation capabilities. In order to share a common baseline vehicle simulation, whose software components will ultimately come from different program participants, a standard set of variable names, sign conventions, model architecture, and data table formats must be agreed upon. One possible solution would be to adopt the conventions, architecture, and formats of one participant (requiring all sites to rehost and adapt existing routines to accommodate these formats). This solution may not be the best, however, from a standpoint of long-term acceptable conventions and architectures for other similar teaming efforts.

In an attempt to begin to understand the differences in simulation software practices at various simulation facilities, and to search for possible solutions, a questionnaire was mailed to each of the members of the AIAA Flight Simulation Technical Committee, as well as selected additional simulation organizations, requesting information about software methods practiced at each organization. 74 questionnaires were mailed; as of this writing, 21 have been completed and returned; four were returned without completion. Two of the completed surveys were determined to be inappropriate for inclusion due to the nature of the responding facility. Responses were received from government (DOD and NASA) flight research and test facilities, simulation software contractors, as well as several universities. Major airframe developers in general failed to respond to the questionnaire, unfortunately.

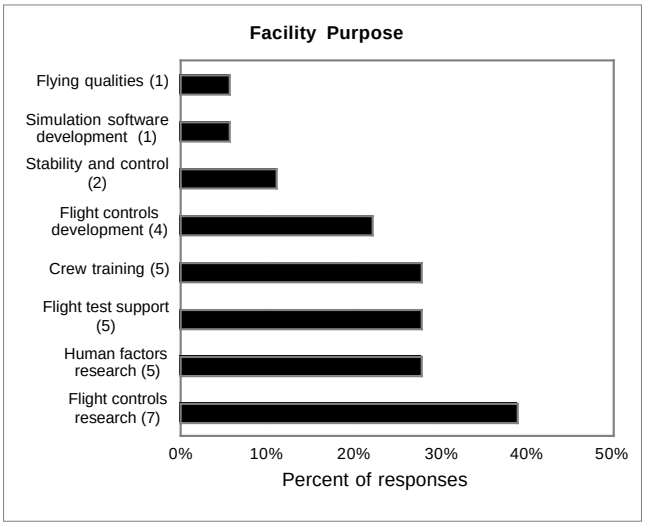
The questionnaire was a ten-page form designed to be filled out in under four hours; it invited the correspondent to provide internal documentation in lieu of detailed responses to particular questions. It was available electronically, as well, via the Internet.

Specific areas addressed were facility description, dynamic model software formats, facility analysis capabilities, and software development methods. Additional information about data standards, variable naming conventions, and a short facility history was requested.

Results

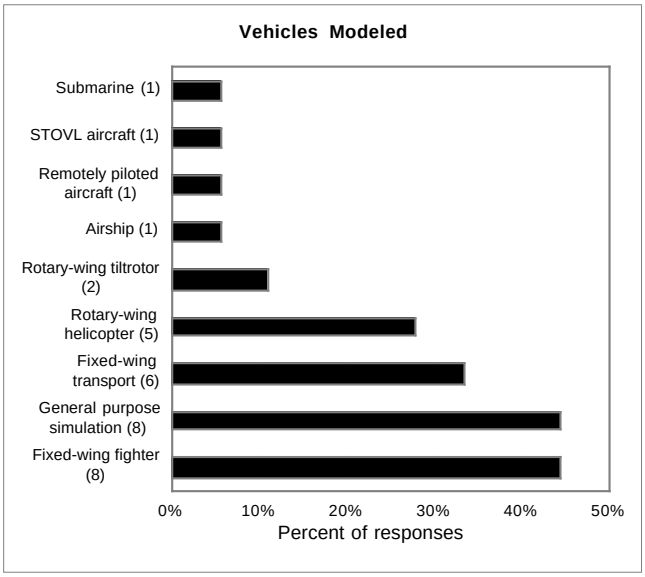
Facilities represented

In response to "Please describe the primary purpose of the simulation models at your facility (e.g. engineering analysis, crew training, subsystem development, flight research)," eight major groupings of "facility purpose" were apparent (some respondents indicated their site had more than one purpose):



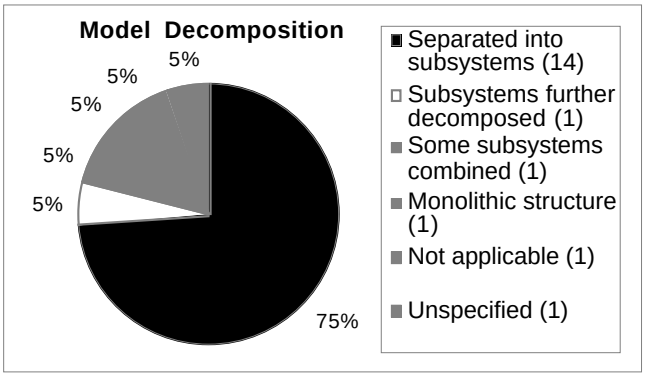
* Senior Research Engineer, Flight Dynamics and Controls Division, Dynamics and Control Branch, Member AIAA

In response to "Please indicate what category and class of flight vehicle your facility represents in its simulation models (e.g. rotor wing helicopter, fixed-wing fighter, fixed-wing transport)," the following vehicle types emerged (some responses indicated more than one primary type):

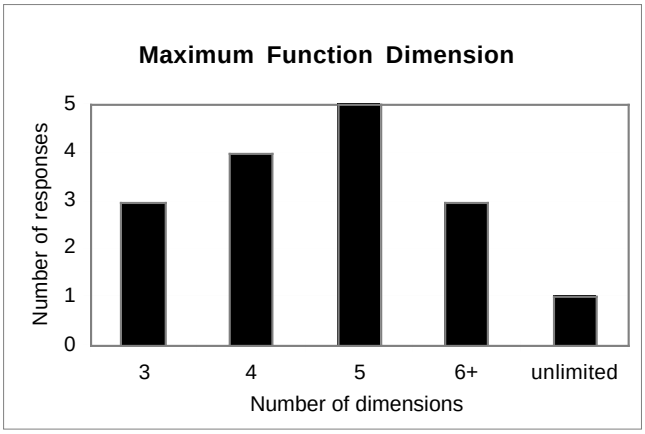


Modeling and Data Representation

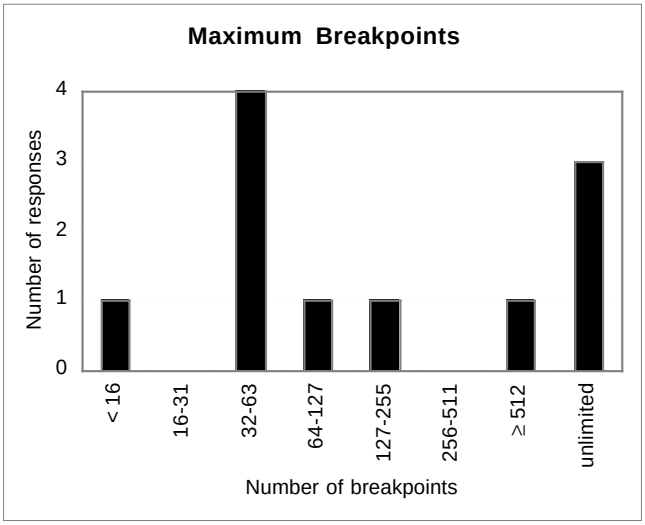
Shown below is the response to "How does your organization prefer to partition the various parts of the flight vehicle model? For example, do you write separate aerodynamic, propulsion, mass/inertia, landing gear, etc. software routines?":



When asked, "Does the non-linear function representation methods used by your organization provide linear interpolation in multiple dimensions?" eighteen respondents answered affirmatively; one did not indicate a response. A follow-up question, "If so, what is the maximum number of dimensions supported?" yielded this distribution of answers:



An additional follow-up on data interpolation, "What is the maximum number of breakpoints in each dimension?" yielded the following distribution:



This question was posed next: "[Are] Spline fits [used]? If so, what order?" Four of the respondents indicated that splines were used: one reply indicated that a second-order spline was used, one indicated third-order, and two indicated splines were available, but no limit was specified.

Another follow-up question was: "[Is] Polynomial representation [used]? If so, what order?" Four responses indicated polynomial expressions for non-linear functions were used, with a fourth, fifth, and sixth-order polynomial indicated on one response each; one response indicated no limit to the order of polynomials used to represent non-linear functions.

Nine respondents indicated some form of extrapolation could be used beyond the boundary of a function table; fourteen indicated a "hold-constant last value" approach was customarily used when an independent variable exceeded the bounds of a function table. Only two facilities provide some indication that the "data base boundary [has been] exceeded" to the user under these conditions.

The survey requested information about the capabilities of the respondent's equations of motion calculations. A checklist followed the request, "Please indicate features you have found necessary for equations of motion calculations." The number of organizations indicating provision of each feature is tabulated below (one respondent did not answer this question):

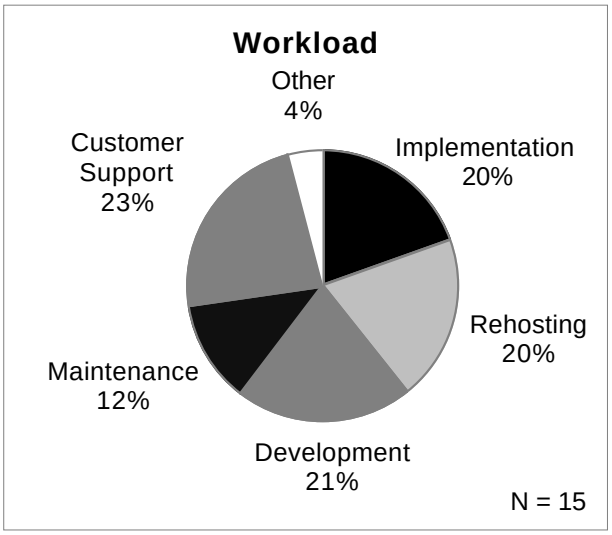
% (N=18)	Equations of motion provisions
100.0	English units of measure supported
72.2	Quaternions used for angular state integrations
66.7	Integrations performed in body axes
66.7	Flat earth assumed
61.1	Allowance for rotating machinery
44.4	Round earth assumed
44.4	Asymmetric interia tensor allowed
33.3	Uses specialized or tuned integration scheme
33.3	Oblate earth effects modeled
33.3	Aeroelastic calculations
27.8	Follows ANSI/AIAA R-004-1992
22.2	Integrations performed in inertial axes
22.2	Utilizes variable time step integration scheme
22.2	Rotating earth/atmosphere modeled
11.1	ISO/Metric units of measure supported

Resource Allocation

The survey included questions regarding human resource allocation. The first question was:

> For the last two or three years, please estimate what percentage of your staff's time is spent in the following activities:
>
> - Implementing new vehicle dynamic models (e.g. writing, debugging, and documenting new aerodynamic, propulsion, or control subsystem models)?
> - Rehosting existing vehicle dynamic models into your facility's real-time architecture (this could include upgrading off-line analytical models into real-time simulation models)?
> - Developing new generic simulation capabilities, such as incorporating new hardware and writing new analysis software?
> - Maintaining existing simulation software?
> - Supporting your customers (both internal and/or external customers) through simulation runs and tests?
> - Other

Fifteen surveys were returned that included answers to this question. Although the responses showed large variations of resource allocation in each of the six categories, the averaged percentages were tallied:



In response to the question, "How many work-months would you estimate are required to properly implement a typical new simulation, starting with whatever source data you typically receive, until the simulation is available for productive use?" the average response was 11.0 work-months; the minimum average effort estimated (for a six degree of freedom non-linear aircraft model) was 4.3 work-months; the maximum estimate given was 24 work-months.

Validation and Analysis Capabilities

A question about model validation techniques was posed: "Please indicate what techniques are used at your facility to validate a new or rehosted simulation model (e.g. flight test time history comparison, analytic model time history comparison, frequency domain methods, subjective pilot opinion)." The percentage of sixteen replies indicating utilization of various techniques is given below:

% (N=16)	Validation technique
93.8	Subjective pilot opinion
75.0	Trim state comparisons with preflight predictions
68.8	Trim state comparisons with flight test data
68.8	Time history comparison with flight test data
62.5	Time history comparison with preflight predictions
56.3	Frequency comparison with flight test data
37.5	Frequency comparison with preflight prediction
31.3	Static checks at nonequilibrium initial conditions

One facility volunteered three additional techniques: subsystem static and dynamic checks, data table plots, and calculation of stability derivatives for comparison with predicted values.

A set of questions pertaining to the analysis capabilities of each simulation site were asked. The tabulation below shows the number of respondents who indicated their simulation software provided these basic analytical capabilities:

% (N=17)	Basic analysis capability
88.2	Six degree of freedom (or more) trim capability (e.g. trim in turns)
82.4	Three degree of freedom (e.g. longitudinal) trim
52.9	Support for both real-time and batch access with same executable image
52.9	Linear model extraction capability for basic aircraft/rotorcraft aerodynamic states
47.1	Limited parameter identification capability
29.4	Linear model extraction capability for actuator, engine, and/or control system states
17.6	Optimal trim capability for more than six degrees of freedom

The questionnaire provided an option to describe additional capabilities in addition to the ones listed in the survey. In response to the question, "Beyond these, what other important analysis capabilities are available?" the table below lists the capabilities that were cited, and the number of simulation facilities that featured these capabilities:

% (N=13)	Other analysis capability
38.5	Automatic test maneuver generation
23.1	Time history playback
15.4	Frequency domain analysis
15.4	Aircraft hardware in-the-loop
7.7	Time history matching tools
7.7	Sensitivity analysis tools
7.7	Black box recording
7.7	Time delay measurement
7.7	Data manipulation tools
7.7	Parameter identification tools
7.7	No additional analysis capabilities

In response to an inquiry of "What new vehicle simulation analysis capabilities are planned for your software?" the following responses were tabulated:

% (N=11)	Planned capabilities
45.5	No additional capabilities planned
18.2	Parameter identification
9.1	Experimenter's station
9.1	Frequency domain analysis
9.1	Linear model extraction
9.1	Performance improvement
9.1	Time history playback

A final question, "What new analysis capabilities are desired of future flight simulations?" resulted in these responses:

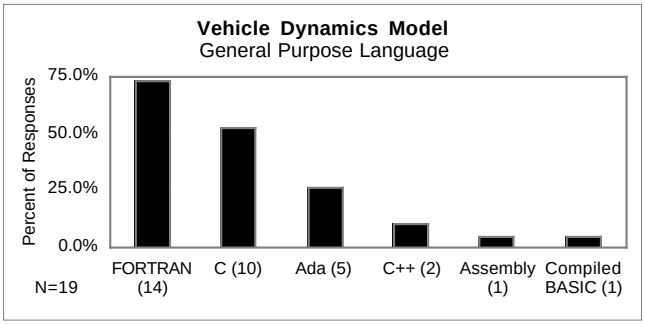
% (N=10)	Desired capability
30.0	Time history playback
20.0	Block diagram programming environment
20.0	Parameter identification tools
10.0	Faster validation tools
10.0	Frequency domain analysis
10.0	Modeling structural modes
10.0	Library of standard model components
10.0	No additional capabilities desired

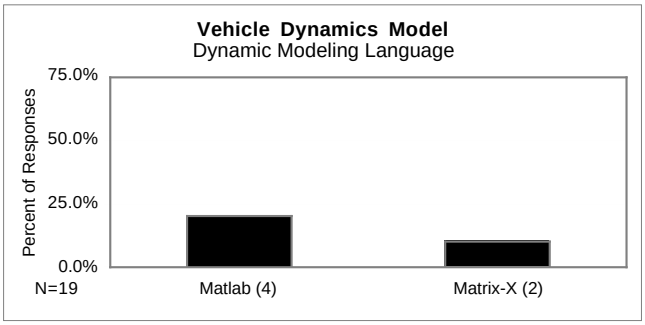
Software Methods

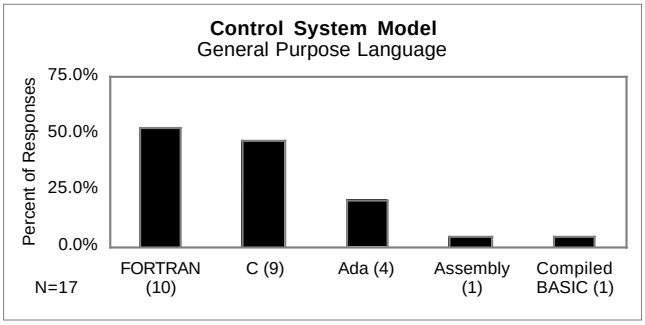
Modeling Languages

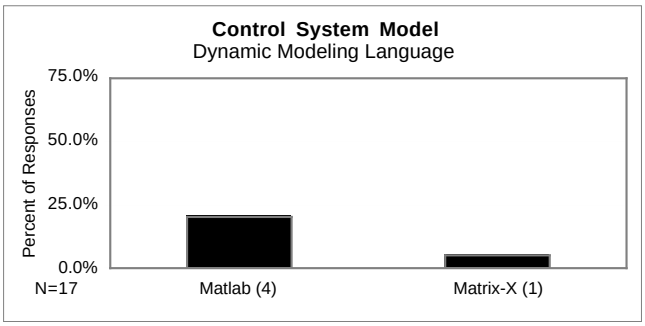
The questionnaire then asked two questions about software tools and methods used in the development of flight simulation systems. The first question inquired about programming languages: "What type of software does your organization typically use to describe vehicle dynamics, such as aerodynamic, propulsion, and landing gear models?" was followed by two sections: a list of popular general-purpose programming languages, and a list of dynamic modeling languages.

The second question inquired about software used specifically for control system modeling: "What software do you typically use to model the flight control system ?" was followed by the same two lists of general purpose and dynamic modeling languages. Of the 19 respondents that answered this section, the percentage of use of a particular language are tallied below:









One respondent indicated that hardware-in-the-loop was used in place of a software-based control system model, and one respondent used an in-house block diagram language to code the control system model.

Arbitrary variable access (peek and poke)

In response to the query, "Does your simulation software provide a mechanism for access to arbitrary variables at run time (sometimes referred to as 'peek and poke' capability)?" 15 respondents (79 %) indicated some form of this capability is provided to the simulation operator while four sites (21 %) did not provide this capability. A follow-up question allowed the

respondent to indicate how this was accomplished. Five facilities used the compiler's debugging symbol table, and ten sites used an on-line data dictionary to provide addresses.

Parameter passing

In response to the question, "Please indicate which method of passing parameters between major software modules is utilized at your facility," followed by a list of four suggested responses as well as blank lines to describe a method not listed, the 18 respondents that completed this section of the survey provided responses as follows (more than one choice was allowed):

- 56 % Used FORTRAN COMMON blocks to pass parameters between modules
- 50 % Variables, or sets of variables, are passed in subroutine/function call
- 39 % Header files are used to share global variable declarations
- 39 % Precompiler & predefined data dictionary is used
- 33 % "Include" directive is used to insert global memory definitions

Software configuration control

Of three choices suggested after the request "Please check all that apply to simulation code development at your facility" the 17 respondents that answered this question made the following choices (more than one choice was allowed):

- 94 % Standard software routine/library safeguarded by specific individual(s)
- 53 % Use rcs/sccs or other source code control tool
- 35 % Formal software verification/walkthrough procedure commonly followed

One respondent volunteered that a software configuration control board was used to safeguard production configurations.

Software coding standards

When asked, "Does your organization follow any formal software coding and documentation standards?" the following distribution of responses emerged:

Choice	N = 18	Percent
Yes	10	56 %
No	8	44 %

In response to "Does your organization routinely write ANSI compliant software (FORTRAN, C, and/or Ada)?" the following responses were tallied:

Choice	N = 18	Percent
Yes	11	61 %
No	7	39 %

In response to "Does your organization routinely develop code using the methods described in MIL-STD-2167?" the following responses were tallied:

Choice	N = 18	Percent
Yes	3	17 %
No	15	83 %

Dynamic element subroutine/function library

To the query, "Does your organization provide a library of dynamic element models (lead-lags, transient-free switches, etc.) for all simulation models?" the respondents answered:

Choice	N = 18	Percent
Yes	9	50 %
No	9	50 %

Two respondents provided a list of their "standard" dynamic element models.

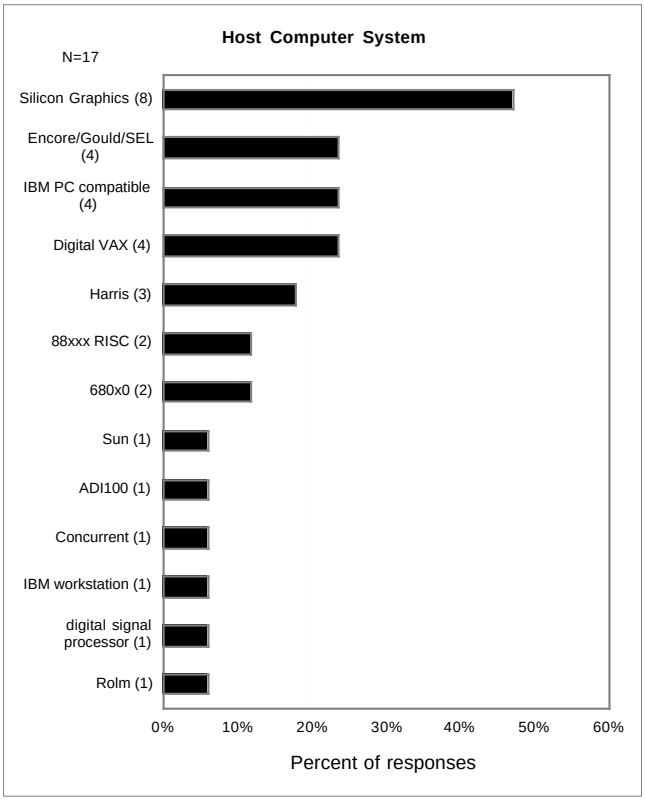
Software commonality

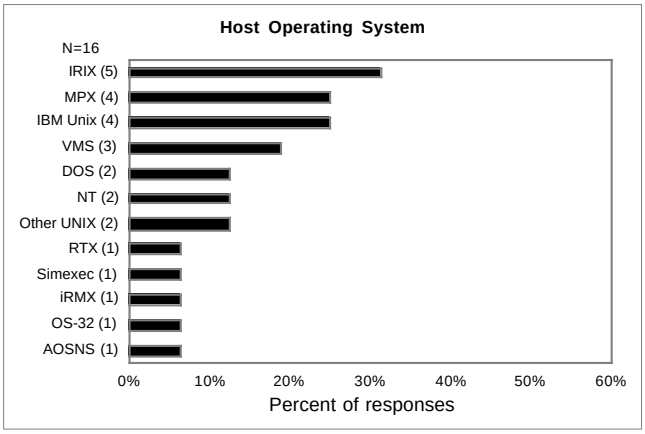
In an attempt to discover the extent to which software is reused at a typical simulation facility, the questionnaire asked, "Does your organization use a single software architecture for more than one flight vehicle model? For example, is there a single executive routine that calls a standard set of flight vehicle-specific subroutine or subroutines?" The choice selections were as follows:

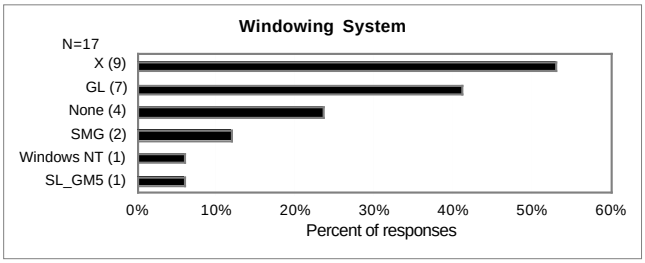
Choice	N=18	Percent
Standard architecture for most simulations	11	61 %
Some commonality	3	17 %
Little commonality (few standard routines)	4	22 %

Real-time Hardware & Operating System

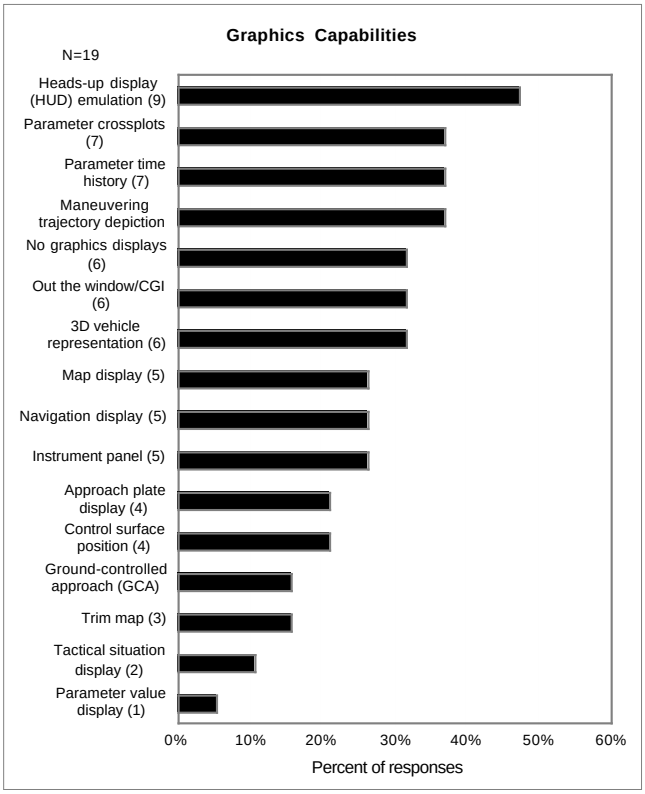
In response to questions about host computer system hardware, operating system and user interface, the completed questionnaires indicated the following responses (several sites indicated more than one choice):







All but five percent of the responses indicated some type of computer-generated graphics display was utilized to support simulation. The use of this graphical capability was then requested: "Please describe any special graphical capabilities, such as trajectory visualization, trim maps, strip chart emulation, parameter cross-plots, vehicle attitude depiction (spin graphics), air combat situation display, HUD emulation, control surface position depiction, or any other 3D animation feature." The responses are tallied below:



Supplemental Information

The respondents were asked to supply additional documentation about their facility's format for non-linear function data and time-history data; a request was also made to provide any documentation on the convention used for variable names in simulation code. In response to these requests, nine facilities either provided or described their function data representation formats; three facilities provided or indicated their time history data format; only two facilities provided a list of typical variable names.

Eleven facilities provided, in response to a request, a short narrative of the origins of their simulation software architecture. Nine facilities developed their software in-house, one facility procured their software via contract, and one facility co-developed their software with contracted assistance.

Discussion

Although the sample size of this survey is fairly small, it is believed to be representative of the aircraft flight simulation community at large, due to the fairly diverse types of facilities and vehicle types sampled, with one notable exception: perhaps due to a concern over compromise of proprietary information, no major airframer responded to the survey.

All but one respondent indicated that their simulation facilities supported real-time, pilot-in-the-loop simulations.

Modeling and Data Presentation

The majority of simulation software architectures seem to decompose the vehicle dynamics into separate subsystem models. This is not a surprising result, given the complexity of most engineering-quality aircraft simulations.

The average simulation architecture supports at least 4 dimensions in non-linear function table, with at least 32 breakpoints per dimension; a number of sites support an unlimited number of breakpoints in each dimension. Smoothed interpolation of non-linear function data is not typically available, however. A typical function interpolation scheme holds the last value for the dependent variable when an independent variable exceeds the specified range, but does not provide an indication that the boundary of valid data had been exceeded.

Almost all respondents use English units, reflecting American aerospace practice. Most simulation sites assume a flat earth, use quaternions for angular state integrations (to avoid the singularity at $\pm 90°$ pitch attitude), account for rotating mechanisms, and perform integrations in body axes. About one-third account for aeroelastic effects; less than one-third follow the 1992 ANSI/AIAA Recommended Practice[1] for sign conventions and axis systems.

Resource Allocation

Over half the human resources at a typical flight simulation facility is spent supporting customers, improving the facility, or maintaining existing code; one-fifth of the resources are spent rehosting an existing simulation models. The average resources spent in implementing a new simulation model is 11 staff-months.

Validation and Analysis Capabilities

The typical real-time simulator facility uses trim shots, dynamic

checks, and piloted evaluations to validate a simulation model. Frequency domain comparisons are less used than time domain comparisons. Approximately one-half of all simulator facilities provide linear-model extraction capability for the rigid-body open-loop plant. Also, one-half allow both batch- and real-time operation of the same simulation executable image. Several facilities have developed an automated test maneuver autopilot capability.

Software Methods

FORTRAN is presently the mainstay of flight simulation software, with C, Ada, and C++ being used less frequently. Less than one-third of the facilities surveyed use specialized dynamics modeling language for vehicle dynamics. FORTRAN and C are typically used for modeling the control system, with Ada a distant third choice. Specialized dynamics languages are used in less than one-third of the facilities for control system models.

Most simulator facilities designate software librarians or other personnel to safeguard important code. Approximately half employ computer-based revision or source-code control system software to maintain configuration control; less than half employ a formal software design review or require formal software documentation. Just over half the sites surveyed write ANSI compliant software. Only three facilities follow the requirements of MIL-STD-2167, the Department of Defense formal software development and testing protocol.[2]

Only half the facilities surveyed maintain a library of standard dynamic element software routines. While most facilities provide a standard architecture for all simulations within the facility, a substantial portion (39 %) do not.

Real-time Hardware & Operating System

A wide variety of host computers and operating systems are used in the simulation laboratories across the country, with a mix of traditional older 32 bit complex-instruction set computers and newer 64-bit reduced-instruction set workstations; personal computers have a foothold as well. Variants of UNIX are used at over two-thirds the simulation facilities surveyed, with off-the-shelf (not specialized for real-time operation) UNIX utilized in half of the facilities. X-windows is utilized at over one-half the facilities surveyed as part of the operator interface; the majority of simulation facilities utilize or plan to utilize graphical workstations, with heads-up display (HUD) emulation, trajectory depiction, strip chart displays, and parameter cross-plots leading the list of graphics applications.

Concluding Remarks

This survey reveals an industry in transition: faster and cheaper workstation and desktop computers running UNIX are supplanting the traditional large mainframe machine with proprietary operating system. Modern structured and object-oriented computer languages are appearing alongside FORTRAN as the language of choice to reduce life cycle cost. Modern software configuration control schemes are being put in place to safeguard the sizable software investment.

What is also apparent is the diversity and incompatibility of most simulation software architectures: a wide range of function interpolation capabilities, axis systems, variable naming schemes, and validation techniques are entrenched at each facility, due no doubt to the independent development and evolution of these facilities. Thus, flight vehicle models created at one facility must be modified substantially to run at another facility. Indeed, the procurement of newer host computers may require substantial recoding, especially if a new language or operating system is employed. This incompatibility is reflected in the substantial portion of the simulation software workforce (20 %, or one-fifth) typically engaged in rehosting existing models. The current trend in industry teaming for aerospace vehicle development will cause an incompatibility to become more onerous, as will the emergence of desktop-based simulations.

Additional work in the area of standards for aircraft modeling software is needed to reduce the incompatibility between simulation facilities and increase the productivity of this technical art.

References

[1] American Institute of Aeronautics and Astronautics, sponsor: Recommended Practice: Atmospheric and Space Flight Vehicle Coordinate Systems, ANSI/AIAA R-004-1992, February 1992.

[2] U.S. Government Printing Office: Defense System Software Development, MIL-STD-2167A, February 1988.