THE GENERIC SIMULATION EXECUTIVE AT MANNED FLIGHT SIMULATOR

James Nichols
Naval Air Warfare Center Aircraft Division
Patuxent River, Maryland

1994

Abstract

The Manned Flight Simulator (MFS) at the Naval Air Warfare Center Aircraft Division (formerly the Naval Air Test Center) was created to provide rapid response to a wide range of US Navy simulation requirements. The necessity to simulate any aircraft in the US Navy inventory stimulated the idea of creating "roll-in, roll-out" simulation bays that would accept any cockpit having standard geometric and electrical interfaces. The capability to use any cockpit at any simulation bay in turn led to the need for a flexible and generic software package for simulating any airframe. The Controls Analysis and Test Loop Environment (CASTLE) executive allows the user to easily generate and operate an aircraft simulation, while also providing a very powerful set of tools for simulation development and engineering analysis. Although the CASTLE package was originally designed to operate on Digital Equipment Corporation (DEC) machines using the VMS operating system and DEC screen management software, recent developments include a MOTIF-based window interface environment and compatibility with the UNIX operating system. The CASTLE package is being proposed as a starting point for a standard airframe simulation package to satisfy US Navy requirements.

TO WHOM IT MAY CONCERN:

Here is a copy of Professional Papers written by various people here at the Naval Air Warfare Center Aircraft Division. It was requested that a copy of each of the professional papers be sent to DTIC for retention.

If you have any questions, please contact Dorothy Reppel, 326-1709 or (301) 826-1709.

P.S. All the enclosed papers have been cleared for public release.

THE GENERIC SIMULATION EXECUTIVE
AT MANNED FLIGHT SIMULATOR

James Nichols*
Naval Air Warfare Center Aircraft Division
Patuxent River, Maryland

Abstract

The Manned Flight Simulator (MFS) at the Naval Air Warfare Center Aircraft Division (formerly the Naval Air Test Center) was created to provide rapid response to a wide range of US Navy simulation requirements. The necessity to simulate any aircraft in the US Navy inventory stimulated the idea of creating "roll-in, roll-out" simulation bays that would accept any cockpit having standard geometric and electrical interfaces. The capability to use any cockpit at any simulation bay in turn led to the need for a flexible and generic software package for simulating any airframe. The Controls Analysis and Test Loop Environment (CASTLE) executive allows the user to easily generate and operate an aircraft simulation, while also providing a very powerful set of tools for simulation development and engineering analysis. Although the CASTLE package was originally designed to operate on Digital Equipment Corporation (DEC) machines using the VMS operating system and DEC screen management software, recent developments include a MOTIF-based window interface environment and compatibility with the UNIX operating system. The CASTLE package is being proposed as a starting point for a standard airframe simulation package to satisfy US Navy requirements.

* Aerospace Engineer, Member AIAA

Introduction

Traditionally, the simulation community has been plagued with widely varying requirements that result in numerous simulation architectures. Although meeting the specifications for one project, these architectures usually cannot be reused without considerable time and effort. Some facilities have one simulation package for the pilot-in-the-loop task, and completely different environments for "desk-top" engineering analysis and other tasks. In some cases a significant effort may be required to take a simulation of the same airframe from one environment and re-host it in another. One common example of this is taking an engineering analysis simulation and moving it to the piloted cockpit facility. Although many cases like this may stem from computing platform incompatibilities, the lack of standard software for the simulation executive plays a major role as well.

Another long-standing practice has been to take an existing airframe model, copy it, and modify it as required to generate a simulation of another aircraft. This is certainly a valid approach, but results in having to maintain several sets of code that do essentially the same thing. The time and money constraints that commonly accompany simulation projects dictate that a swift and efficient method be used to develop a new simulation without the software maintenance nightmares.

The CASTLE package resolves both of these issues by providing an environment that already contains the "shell" of modules necessary to simulate a simple linear model of an airship up to a highly non-linear model of the most complex aircraft under development.

The CASTLE "Shell"

The CASTLE package consists of several parts. These are the CASTLE "shell" modules, the simulation development tools, and the airframe-specific modules (provided by the simulation developer). The shell modules include the user interface routines, the variable accessing tools, the engineering analysis packages, the airframe executive routines, and the external communications routines. Figure 1 shows the hierarchy of the CASTLE components.

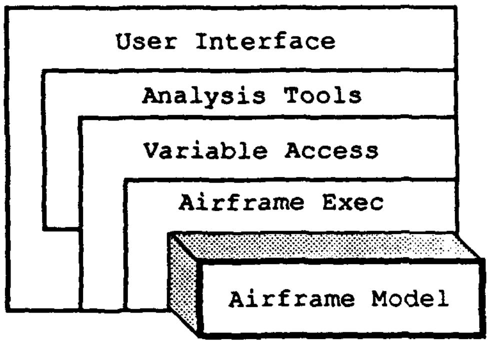

Figure 1: CASTLE Component Hierarchy

The FORTRAN language is used for the airframe executive and is the expected language for the airframe-specific modules. The airframe model itself may be programmed in another language, such as ADA, but interface packets must be created to communicate between the FORTRAN equivalence statements of the CASTLE routines and the other software environment.

User Interface Routines

The user interface routines are intended to provide a user-friendly graphical environment for controlling a CASTLE simulation. The routines communicate to the body of the CASTLE code through several shared memory blocks, so that other user interface packages may be "plugged in" instead of the standard MOTIF windows environment. This method also allows the user interface package to run as a separate task from the airframe executive if desired, and allows commands to come from external sources such as command files. The ability to use command files to submit CASTLE images to batch jobs is desired for computationally intensive engineering analysis tasks. There are currently two forms of the user interface, one using the DEC screen management system, and the other using the MOTIF window environment. Figure 2 shows the communication paths between the user interface routines and the CASTLE utilities.

The user interface environment is designed in such a way that adding new facilities is relatively simple. The user must supply a set of routines that define the new screen format, fill the screen buffer as required, and take data from the screen buffer and load it back into the CASTLE tables. The new facility command is added to a command table and the required "action" routines are created for the CASTLE airframe executive. The use of standard templates can greatly simplify the task.

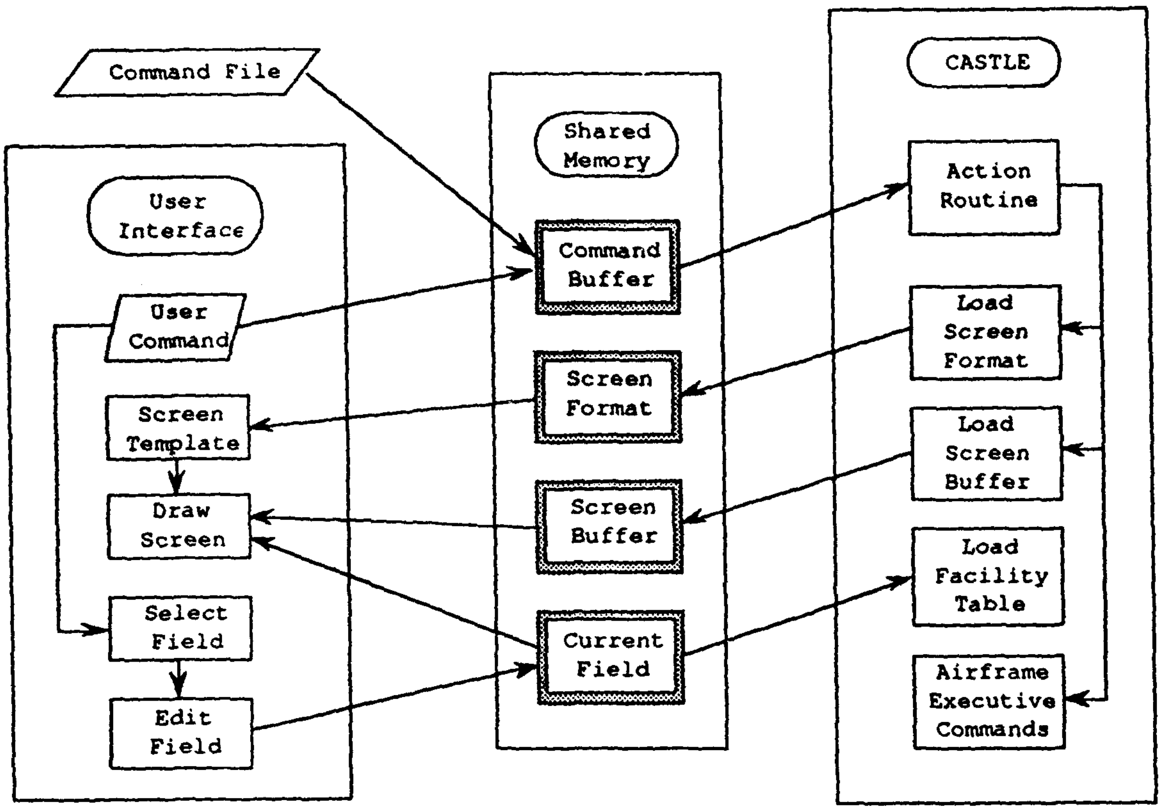

Figure 2: User Interface Connection to CASTLE

Variable Accessing

The key to the flexibility of the CASTLE code lies in the method used to access variables. When the user specifies a variable to be acted upon, the ASCII string is matched to an entry in a file that contains common block definitions. The location within the common block is noted, and the address of the variable is calculated based on the common block address, the base address of the variable within the common block, and any array offsets requested. This methodology avoids the pitfalls of specifically coding variables, and allows the user almost unlimited power to specify variables for trimming and other tasks. The variable address was previously found by retrieving it from the symbol table created by the VAX VMS debugger, but translating a symbolic debugger symbol table is by nature a very machine-specific task and was abandoned.

Analysis Tools

The primary engineering analysis tools that are built into the CASTLE package are the Maneuver Function Generator (MANGEN), the trimming facility, the Linear Model Extraction (LME) utility, and the Simulation Checking using an Optimal Prediction Evaluation (SCOPE) package. Digital time history data is stored by data sets that point to locations in a large data buffer. As many as 50 different time histories may be stored simultaneously for comparison or use by the analysis tools. Figure 3 describes the data storage structure used internally in CASTLE. The digital data may be saved to external files in a variety of standard formats, as well as formats defined by the user. The digital data may also be stored externally during a pilot-in-the-loop session by passing it through shared memory to a real-time output routine, allowing virtually unlimited storage capacity.

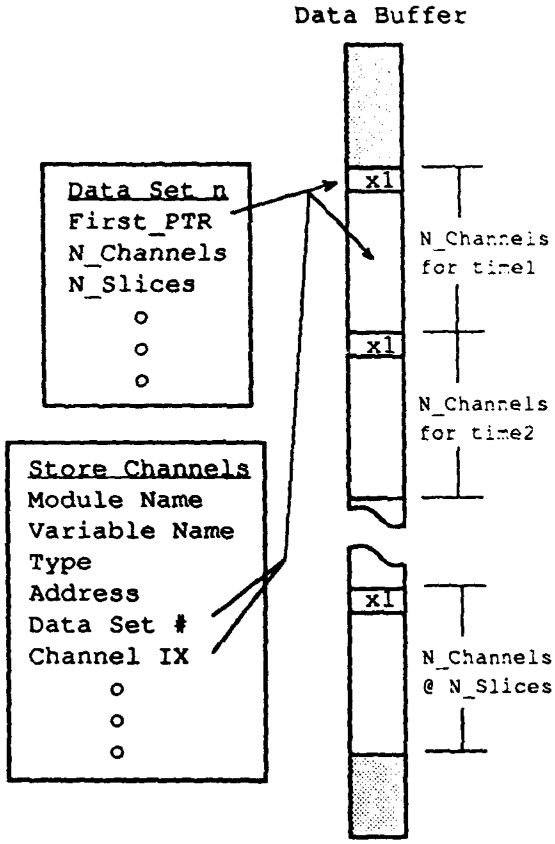

Figure 3: CASTLE Time History Storage

Maneuver Function Generator

The Maneuver Function Generator (MANGEN) allows the user to drive any available variable with a pre-defined function or combination of functions. MANGEN will also accept digital time histories as driving inputs. It is mainly intended to serve as a substitute for real-time pilot control inputs. Another utility under development is a preliminary version of a virtual pilot, which will be used as a true aircraft maneuver generator. This will allow the user to define a target flight condition and have the aircraft fly to the specified condition and execute the requested maneuver. It could also be used for commanding maneuvers such as "S-turns", wind-up turns, and level accelerations.

CASTLE Trim Facility

The CASTLE trim facility perturbs the user-defined trim controls to drive the user-specified state derivatives to target values for a requested set of initial conditions. The trim routine may use any available variable for a control or state derivative, and thus is extremely versatile. It may be used to trim out asymmetric store loadings, or trim to a flight path, or trim in any steady state flight condition. Some pre-defined conditions that may be selected include constant-rate turns, pull-ups, and push-overs.

Linear Model Extraction

One of the most useful utilities in CASTLE is the Linear Model Extraction (LME) facility. It allows the user to define a linear model structure using ASCII simulation variable name strings for states, inputs, state derivatives, and outputs. The A, B, C, and D matrices that pertain to the equation set:

$$\dot{\mathbf{x}} = [A]\mathbf{x} + [B]\mathbf{u}$$
$$\mathbf{y} = [C]\mathbf{x} + [D]\mathbf{u}$$

where:

$\dot{\mathbf{x}}$: state derivative vector
$\mathbf{x}$: state vector
$\mathbf{u}$: input vector (controls,etc)
$\mathbf{y}$: output vector

are determined by perturbing the states and inputs from a pre-defined state and measuring the effect on the state derivatives and the outputs. Several perturbations are used with varying step sizes to get the best approximation of the partial derivative for each matrix element. The output results may be written to a MATLAB® "M" file. The LME utility can be used to generate a linear model of any portion of the airframe model by using the pre-defined model selection types or by the judicious use of the airframe executive module enabling flags.

Simulation Checking Using an Optimal Prediction Evaluation (SCOPE)

One of the more frustrating tasks in developing an airframe simulation is getting it to perform like the actual article. The SCOPE package allows the user to drive the simulation with time history data and make a qualitative and quantitative comparison of the simulation output to the input time history. It can be used to get an estimate on the initial conditions biases as well as average biases on the total forces and moments that will minimize the error between the input and output data. The biases on the forces and moments merely identify which portions of the model need attention. The Low Order Estimation Tool (LOST) and LOST eXtension (LOSTX) utilities are used to identify, build, and test corrections to the model without recompiling code. The LOSTX utility may be used to incorporate these correction factors without corrupting an established model.

Airframe Executive

The airframe executive consists of the airframe loop executive, the equations of motion, the atmosphere model, ground interface routines, and templates for the airframe model routines. The equations of motion are a highly modified version of the SMART routine used in the NASA Ames BASIC simulation package[1]. The atmosphere model uses the ARDC62 standard model and incorporates the ability to define non-standard day ambient temperature and pressure, as well as density and pressure altitudes. Steady state winds aloft are added to random turbulence and step gusts if desired. A burble model is available for the carrier landing environment, and a turbulence grid generated from Computational Fluid Dynamics (CFD) software is being incorporated for LHA-class ships.

Airframe-Specific Requirements

The airframe developer is responsible for providing airframe-specific modules to satisfy the airframe loop executive. Table 1 lists the procedures that the loop executive will call. If a user-specific routine is not supplied, a dummy routine will be inserted instead. The airframe modules may be coded in any language, as long as the variables are accessible in a common block and documented properly.

COCKPIT	Cockpit controls/switches
IMPORT	External inputs (piloted)
ELEC_SYS	Electrical systems
HYD_SYS	Hydraulic systems
SENSOR	Air data, gyros, etc.
CONTROL	Control laws
SURFACE	Surface actuators
ENGINE	Propulsion systems
AERO	Aerodynamics
WAITIN	Weight, cg and inertias
MISC_FM	Miscellaneous (sling load)
EXPORT	External outputs (piloted)

Table 1: Airframe-Specific Modules

The airframe programmer is free to replace any CASTLE module with a customized version if desired, but the new version must reside outside the baseline CASTLE libraries. If enough interest is generated, the new version may be incorporated in the production CASTLE package. The airframe executive looping routine is sometimes replaced due to specific airframe requirements.

Most simulations use some form of function table lookup for the massive data that accompanies an airframe model. The CASTLE environment encompasses a Function Table Processor tool[2] that takes function data and converts it into FORTRAN functions. The form of the function access is very easy to understand when trying to interpret the top-level code, and looks like:

```
CLALFA = CLALFA_FTP( MACH, ALT )
```

where MACH and ALT are the independent arguments.

If using FORTRAN for a module, the standard CASTLE header should be used and the glossary section filled out appropriately. All "local" variables that will be accessed should be documented in the "LOCALS" section of the glossary. The EDITCOM utility should be used to generate Common Descriptor Files (CDF) for all common blocks specific to the airframe simulation. The CASCOMP pre-compiler should then be used to generate equivalence statements. CASCOMP also uses the CDFs to add the proper variable descriptions to the glossary and to check the type declarations of the "global" variables. The "LOCALS" section is used to create a CDF for the specific module that contains local variables that may be accessed at run-time. The CASTLE coding standards require that all variables used in a procedure be declared explicitly.

AEROPLOT

The AEROPLOT utility is a separate subset of the CASTLE package. It is primarily used to drive the complete aerodynamic model by sweeping selected parameters and observing the output with plot graphics or digital analysis. This is most useful during the initial development phases, as it checks the end-to-end aerodynamic data and catches improperly implemented data functions and equations. It is also good for depicting the aerodynamic coefficients in different axis systems. The effect of a control surface or state on the total aerodynamic model may be observed also. A capability to use AEROPLOT as an engine for generating coefficients not explicitly included in the model is being incorporated as well. The AEROPLOT package uses the standard CASTLE user interfaces and variable accessing tools. The airframe programmer must supply an interface routine between the AEROPLOT executive and the same airframe code that is linked to the CASTLE simulation. Although AEROPLOT was originally developed as an aerodynamic modeling tool, any portion of the airframe simulation may be linked to AEROPLOT. It is an invaluable method for doing end-to-end checks of a subsystem model.

The Near Future

The CASTLE package is continually adding new capabilities in response to the ubiquitous "what if...?" asked by the users. The modular interfaces and coding designs make such expansions a generally straightforward process. One such effort is underway to tightly couple CASTLE and sophisticated analysis tools such as MATLAB® and other packages that perform Parameter Identification (PID) and similar advanced techniques.

Conclusion

The CASTLE simulation environment developed at Manned Flight Simulator represents a considerable effort to satisfy as many simulation requirements as possible while retaining as much modularity and flexibility as possible. As such, CASTLE makes an ideal candidate for a standard simulation package.

References

[1] McFarland, R.E., A Standard Kinematic Model For Flight Simulation at NASA-Ames," NASA CR-2497, January 1975.

[2] Nichols, J.H., "MFSFTP User Guide Version 3.2," Naval Air Warfare Center Aircraft Division, Maryland, 1994.