Software

Generic Environment for Simulating Launch Operations

GEM-FLO (A Generic Simulation Environment for Modeling Future Launch Operations) is a computer program that facilitates creation of discrete-event simulation models of ground processes in which reusable or expendable launch vehicles (RLVs) are prepared for flight. GEM-FLO includes a component, developed in Visual Basic, that generates a graphical user interface (GUI) and a component, developed in the Arena simulation language, that creates a generic discrete-event simulation model. Through the GUI, GEM-FLO elicits RLV design information from the user. The design information can include information on flight hardware elements, resources, and ground processes. GEM-FLO translates the user's responses into mathematical variables and expressions that populate the generic simulation model. The variables and expressions can represent processing times, resource capacities, status variables, and other process parameters needed to configure a simulation model that reflects the ground processing flow and requirements of a specific RLV. Upon execution of the model, GEM-FLO puts out data on many measures of performance, including the flight rate, turnaround time, and utilization of resources. This information can serve as the basis for determining whether design goals can be met, and for comparing characteristics of competing RLV designs.

This program was written by Martin Steele of **Kennedy Space Center** *and Mansooreh Mollaghasemi and Ghaith Rabadi of Productivity Apex, Inc. For further information, contact Mansooreh Mollaghasemi at info@productivityapex.com.*
KSC-12488

Modular Aero-Propulsion System Simulation

The Modular Aero-Propulsion System Simulation (MAPSS) is a graphical simulation environment designed for the development of advanced control algorithms and rapid testing of these algorithms on a generic computational model of a turbofan engine and its control system. MAPSS is a nonlinear, nonreal-time simulation comprising a Component Level Model (CLM) module and a Controller-and-Actuator Dynamics (CAD) module. The CLM module simulates the dynamics of engine components at a sampling rate of 2,500 Hz. The controller submodule of the CAD module simulates a digital controller, which has a typical update rate of 50 Hz. The sampling rate for the actuators in the CAD module is the same as that of the CLM. MAPSS provides a graphical user interface that affords easy access to engine-operation, engine-health, and control parameters; is used to enter such input model parameters as power lever angle (PLA), Mach number, and altitude; and can be used to change controller and engine parameters. Output variables are selectable by the user. Output data as well as any changes to constants and other parameters can be saved and reloaded into the GUI later.

This program was written by Khary I. Parker and Ten-Huei Guo of **Glenn Research Center**. *Further information is contained in a TSP (see page 1).*

Inquiries concerning rights for the commercial use of this invention should be addressed to NASA Glenn Research Center, Innovative Partnerships Office, Attn: Steve Fedor, Mail Stop 4–8, 21000 Brookpark Road, Cleveland, Ohio 44135. Refer to LEW-17674-1.

X-Windows Socket Widget Class

The X-Windows Socket Widget Class ("Class" is used here in the object-oriented-programming sense of the word) was devised to simplify the task of implementing network connections for graphical-user-interface (GUI) computer programs. UNIX Transmission Control Protocol/Internet Protocol (TCP/IP) socket programming libraries require many method calls to configure, operate, and destroy sockets. Most XWindows GUI programs use widget sets or toolkits to facilitate management of complex objects. The widget standards facilitate construction of toolkits and application programs. The X-Windows Socket Widget Class encapsulates UNIX TCP/IP socket-management tasks within the framework of an X Windows widget. Using the widget framework, X Windows GUI programs can treat one or more network socket instances in the same manner as that of other graphical widgets, making it easier to program sockets. Wrapping ISP socket programming libraries inside a widget framework enables a programmer to treat a network interface as though it were a GUI.

This program was written by Matthew R. Barry of United Space Alliance for **Johnson Space Center**. *For further information, contact the Johnson Innovative Partnerships Office at (281) 483-3809.*
MSC-23581

Infrastructure for Rapid Development of Java GUI Programs

The Java Application Shell (JAS) is a software framework that accelerates the development of Java graphical-user-interface (GUI) application programs by enabling the reuse of common, proven GUI elements, as distinguished from writing custom code for GUI elements. JAS is a software infrastructure upon which Java interactive application programs and graphical user interfaces (GUIs) for those programs can be built as sets of plug-ins. JAS provides an application-programming interface that is extensible by application-specific plugins that describe and encapsulate both specifications of a GUI and application-specific functionality tied to the specified GUI elements. The desired GUI elements are specified in Extensible Markup Language (XML) descriptions instead of in compiled code. JAS reads and interprets these descriptions, then creates and configures a corresponding GUI from a standard set of generic, reusable GUI elements. These elements are then attached (again, according to the XML descriptions) to application-specific compiled code and scripts. An application program constructed by use of JAS as its core can be extended by writing new plug-ins and replacing existing plug-ins. Thus, JAS solves many problems that Java programmers generally solve anew for each project, thereby reducing development and testing time.

This software was written by Jeremy Jones and Carl F. Hostetter of **Goddard Space Flight Center** *and Philip Miller and Philip Wheeler of CommerceOne. Further information is contained in a TSP (see page 1).*
GSC-14769-1