Software

Flight Planning in the Cloud

This new interface will enable Principal Investigators (PIs), as well as UAVSAR (Uninhabited Aerial Vehicle Synthetic Aperture Radar) members to do their own flight planning and time estimation without having to request flight lines through the science coordinator. It uses an all-in-one Google Maps interface, a JPL hosted database, and PI flight requirements to design an airborne flight plan. The application will enable users to see their own flight plan being constructed interactively through a map interface, and then the flight planning software will generate all the files necessary for the flight. Afterward, the UAVSAR team can then complete the flight request, including calendaring and supplying requisite flight request files in the expected format for processing by NASA's airborne science program.

Some of the main features of the interface include drawing flight lines on the map, nudging them, adding them to the current flight plan, and reordering them. The user can also search and select takeoff, landing, and intermediate airports. As the flight plan is constructed, all of its components are constantly being saved to the database, and the estimated flight times are updated. Another feature is the ability to import flight lines from previously saved flight plans.

One of the main motivations was to make this Web application as simple and intuitive as possible, while also being dynamic and robust. This Web application can easily be extended to support other airborne instruments.

This work was done by Sarah L. Flores, Bruce D. Chapman, Wayne W. Tung, and Yang Zheng of Caltech for NASA's Jet Propulsion Laboratory. For more information, contact iaoffice@jpl.nasa.gov.

This software is available for commercial licensing. Please contact Daniel Broderick of the California Institute of Technology at danielb@caltech.edu. Refer to NPO-47472.

MPS Editor

Previously, it was time-consuming to hand-edit data and then set up simulation runs to find the effect and impact of the input data on a spacecraft. MPS Editor provides the user the capability to create/edit/update models and sequences, and immediately try them out using what appears to the user as one piece of software. MPS Editor provides an integrated sequencing environment for users. It provides them with software that can be utilized during development as well as actual operations. In addition, it provides them with a single, consistent, user friendly interface.

MPS Editor uses the Eclipse Rich Client Platform to provide an environment that can be tailored to specific missions. It provides the capability to create and edit, and includes an Activity Dictionary to build the simulation spacecraft models, build and edit sequences of commands, and model the effects of those commands on the spacecraft.

MPS Editor is written in Java using the Eclipse Rich Client Platform. It is currently built with four perspectives: the Activity Dictionary Perspective, the Project Adaptation Perspective, the Sequence Building Perspective, and the Sequence Modeling Perspective. Each perspective performs a given task. If a mission doesn't require that task, the unneeded perspective is not added to that project's delivery.

In the Activity Dictionary Perspective, the user builds the project-specific activities, observations, calibrations, etc. Typically, this is used during the development phases of the mission, although it can be used later to make changes and updates to the Project Activity Dictionary. In the Adaptation Perspective, the user creates the spacecraft models such as power, data store, etc. Again, this is typically used during development, but will be used to update or add models of the spacecraft. The Sequence Building Perspective allows the user to create a sequence of activities or commands that go to the spacecraft. It provides a simulation of the activities and commands that have been created.

This work was done by William S. Mathews, Ning Liu, Laurie K. Francis, Taifun L. O'Reilly, Mitchell Schrock, Dennis N. Page, John R. Morris, Joseph C. Joswig, Thomas M. Crockett, and Khawaja S. Shams of Caltech for NASA's Jet Propulsion Laboratory. For more information, contact iaoffice@jpl.nasa.gov.

This software is available for commercial licensing. Please contact Daniel Broderick of the California Institute of Technology at danielb@caltech.edu. Refer to NPO-47237.

Object-Oriented Multidisciplinary Design, Analysis, and Optimization Tool

An Object-Oriented Optimization (O^3) tool was developed that leverages existing tools and practices, and allows the easy integration and adoption of new state-of-the-art software. At the heart of the O^3 tool is the Central Executive Module (CEM), which can integrate disparate software packages in a cross platform network environment so as to quickly perform optimization and design tasks in a cohesive, streamlined manner. This object-oriented framework can integrate the analysis codes for multiple disciplines instead of relying on one code to perform the analysis for all disciplines.

The CEM was written in FORTRAN and the script commands for each performance index were submitted through the use of the FORTRAN "Call System" command. In this CEM, the user chooses an optimization methodology, defines objective and constraint functions from performance indices, and provides starting and side constraints for continuous as well as discrete design variables.

The structural analysis modules such as computations of the structural weight, stress, deflection, buckling, and flutter and divergence speeds have been developed and incorporated into the O^3 tool to build an object-oriented Multidisciplinary Design, Analysis, and Optimization (MDAO) tool.

This work was done by Chan-gi Pak of Dryden Flight Research Center. Further information is contained in a TSP (see page 1).
DRC-010-013