

Joint Polar Satellite System Common Ground System (JPSS CGS) Operational Implementation of Algorithm Changes Using the ADL

Kerry Grant, JPSS CGS Chief Scientist
Raytheon Intelligence and Information Systems, Aurora CO
Gary Metz, IDPS ING/PRO Software Manager
Bryan Henderson IDPS ADL Lead Software Engineer
Paul Siebels, IDPS Deputy Software Manager
Raytheon Intelligence and Information Systems, Omaha, NE



After the successful launch of the JPSS's first satellite, the Suomi National Polar-orbiting Partnership (S-NPP) spacecraft, on Oct. 28, 2011, the Intensive Calibration Validation campaign began in earnest. As Cal/Val proceeds, changes to the science will need to migrate into the operational system. In addition, as new techniques are found to improve, supplement, or replace existing products, these changes will also require implementation into the operational system. In the past, operationalizing science algorithms and integrating them into active systems often required months of work. In order to significantly shorten the time and effort required for this activity, Raytheon has developed the Algorithm Development Library (ADL). The ADL enables scientist and researchers to develop algorithms on their own platforms, and provide these to Raytheon in a form that can be rapidly integrated directly into the operational baseline. As the JPSS CGS is a multi-mission ground system, algorithms are not restricted to S-NPP or JPSS missions. The ADL provides a development environment that any environmental remote sensing mission scientist can use to create algorithms that will plug into a JPSS CGS instantiation.

Advantages: The use of the ADL provides significant time and cost savings during algorithm development and implementation into an operational baseline. It is estimated that by using the ADL, algorithm developers can achieve *10 - 25% cost savings* and that operational conversion/implementation can achieve *25 - 50% cost savings* compared to the cost of typical algorithm development and conversion into an operational baseline.

Usage: The ADL is a tool with a standardized Input-Processing-Output (I-P-O) framework into which algorithm scientists can "plug" their algorithms (Figure 1). Figure 2 illustrates how ADL is used by the algorithm developer to create compatible science code, and by the operationalizing engineer to implement the code in an operational environment. ADL enables the process by providing standardized framework for "Input to Processing" and "Processing to Output" interfaces, standardized toolkits and development GUIs for interfaces, XML GUI editor to define XML product formats that can be processed by the ADL software to auto-generate the input and output software, alleviating the developer of this effort, capability for scientific algorithm developers to design and implement their algorithms on little endian or big endian computing platforms, and "Plug-n-Play" compatibility with operational baseline.

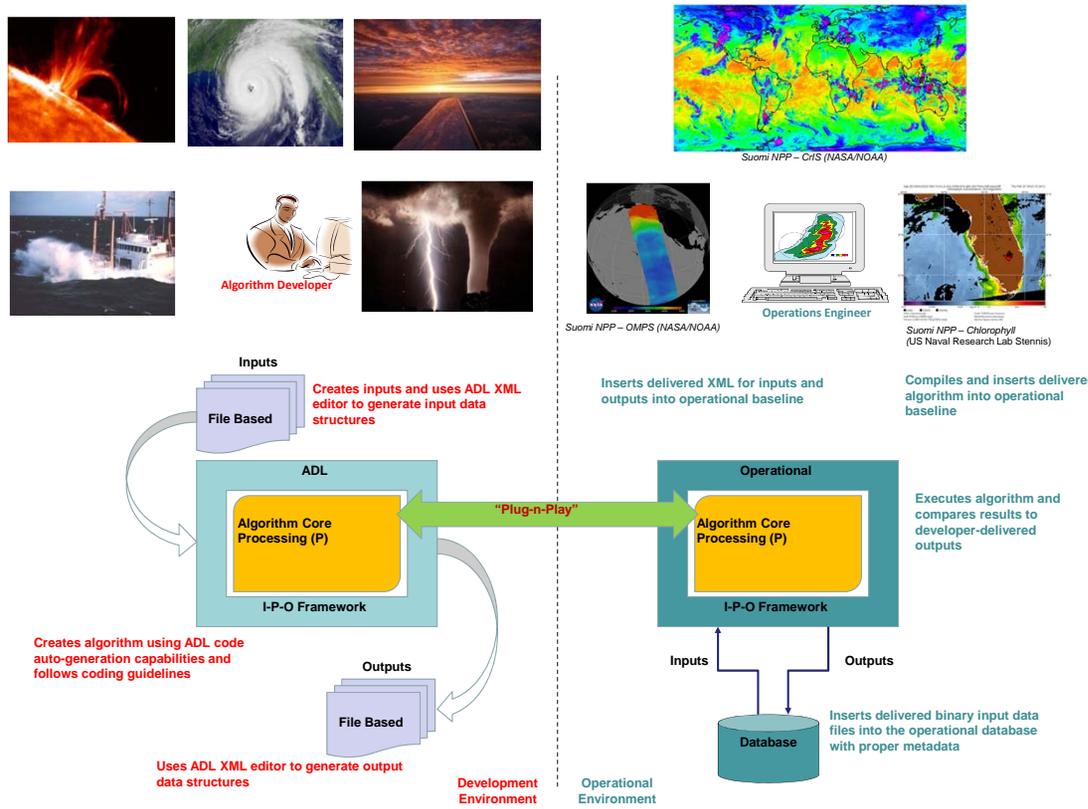


Figure 2 – ADL Concept of Usage

Algorithm Developer

Uses the ADL tool and I-P-O Framework to design and develop the algorithm code in their development environment. Uses the code auto-generation capabilities and ADL coding guidelines to create code compliant with operational standards. Coding can be done on UNIX AIX or LINUX little endian or big endian computing platforms and in C, C++, or Fortran languages.

Operations Engineer

Inserts the delivered algorithm, XML, and inputs into the operational baseline. Completes algorithm conversion and implements operations-unique functionalities. Performs algorithm results comparisons to ensure acceptable results are obtained once the algorithm is fully operational. The operational algorithm can then be returned to the algorithm developer if enhancements or other modifications are needed.

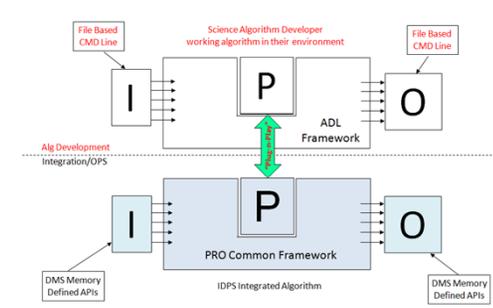


Figure 1 – ADL Isolates Production Software from Science Algorithms

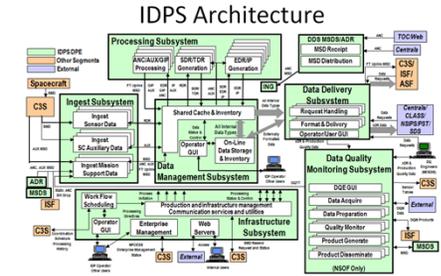
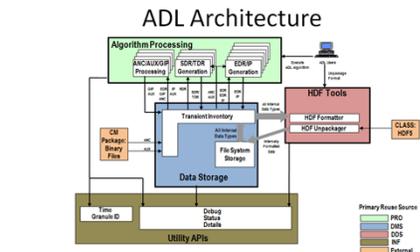


Figure 3 – ADL Simplifies the Algorithm Developer's Work Environment