GOES-R Ground System: Planning for Updates to Level 2 Algorithms

*A. O. Schutte III[1], R. Kaiser[1], **T. Scott Zaccheo**[2], P. A. Van Rompay[3], E. J. Kennelly[2], H. E. Snell[2], W. Wolf[4], S. Sampson[4] , M. Wilson[1]*

**** Presenting***

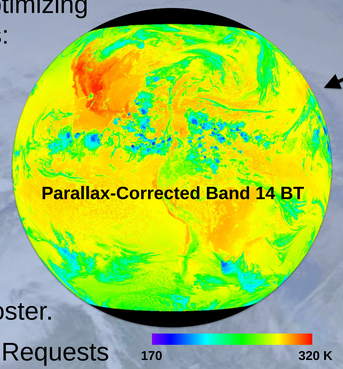
[1]Harris Corp., Melbourne, FL
[2]AER, Inc., Lexington, MA
[3]AER, Inc., Greenbelt, MD
[4]NOAA STAR, College Park, MD

ALGORITHM SPECIAL STUDY

Post-Launch Testing (PLT) of GOES-16 has just started and will include validation of the products from the baseline algorithms. Once this is completed, algorithm updates and new algorithms are expected to be incorporated into the Ground System.

This Algorithm Special Study focused on optimizing the update process with the following topics:

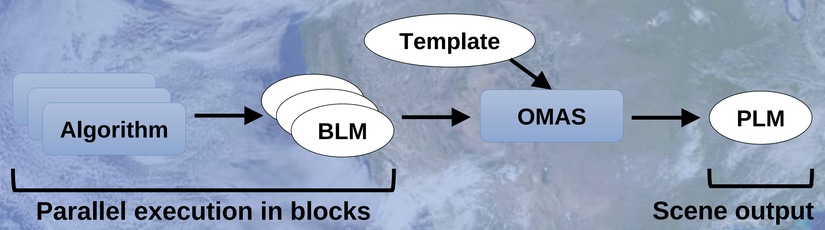
- Algorithm Black-Box Integration
- Metadata Aggregation
- Memory/CPU Footprint
- Algorithm Configuration Database
- Cal/Val Scripts Automation



This study completed in December 2016.
The first 3 topics are summarized on this poster.

The final report gives suggestions for Work Requests and cost / schedule estimates for future algorithm upgrade tasks.

METADATA AGGREGATION (OMAS)



The current Operational Metadata Aggregation Service (OMAS) uses customized computation components, directly interfacing with the algorithm C++ Block-Level Metadata (BLM) objects.

From the Special Study, the recommended upgrade is to create a "Generic OMAS" service which reads XML BLMs and uses an XSLT stylesheet to aggregate the metadata into Product-Level Metadata (PLM). The XML BLMs can be generated by adapting the existing DMI serializer used by all algorithms for their output products. Common metadata like time fields can also be included in this "Generic OMAS".

A prototype for this Special Study was tested on typical statistics: min, max, mean, std. dev. This is an example for computing the mean:

```
<values><xsl:value-of select="sum(to_avg) div
                              sum(to_avg_over)"/></values>
```

2 other options were considered but not analyzed further for this study.

(XSLT = eXtensible Stylesheet Language Transformations)

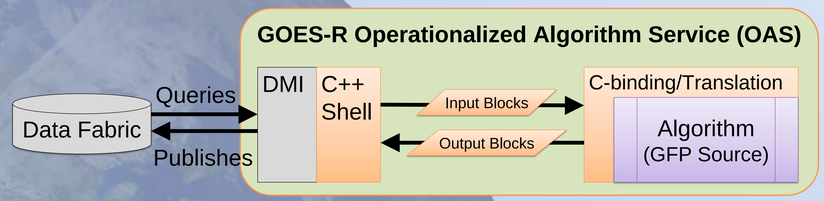
ALGORITHM BLACK-BOX INTEGRATION

An alternative to rewriting scientific code from an Algorithm Working Group (AWG) algorithm into the GOES-R Ground System is to wrap the scientific code and treat it as a "Black Box": inputs are passed in, the AWG code runs the computations, and outputs come out.

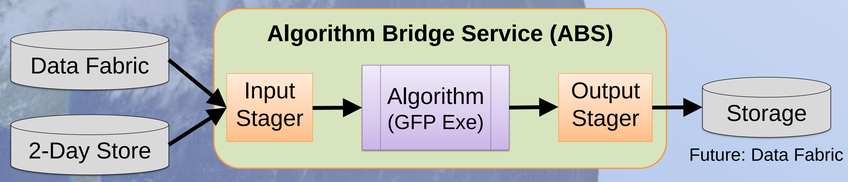
Black-box integration was prototyped using 2 AWG Fortran algorithms:

- Parallax Correction (precedent to GOES-R Rainfall Rate Algorithm)
- Bayesian Cloud Mask (probability-based cloud detection)

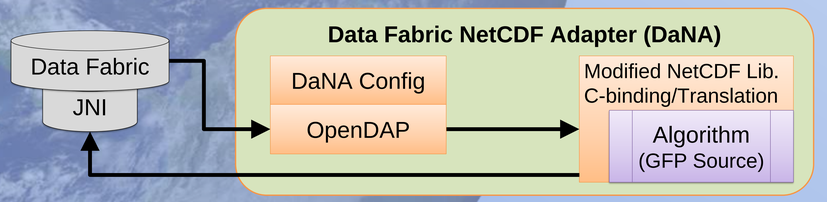
3 approaches were prototyped, as shown below.



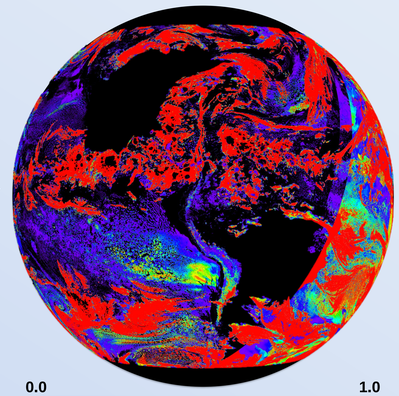
The OAS approach uses the current GOES-R Ground System: Algorithm Shell, Data Model Interface (DMI), and existing infrastructure. The Algorithm Shell converts DMI inputs to Fortran structures and passes them into the Fortran algorithm using C-binding. The outputs are passed back and converted, then written using the DMI. See AMS Poster 246 for details, along with testing using Algorithm WorkBench.

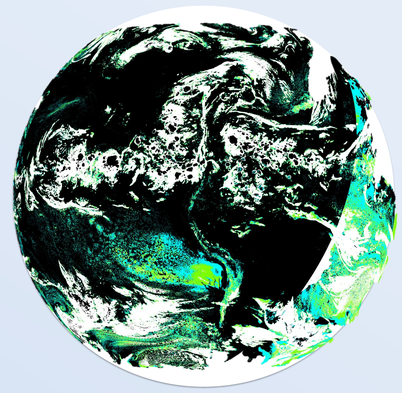


The ABS approach uses the GFP executable, generated using the AWG infrastructure, calling it from the command line. Inputs are staged as NetCDF files, which the executable directly reads and uses to generate its output files. Breaking the scene into blocks is determined by the executable, along with product formatting, and common science code.

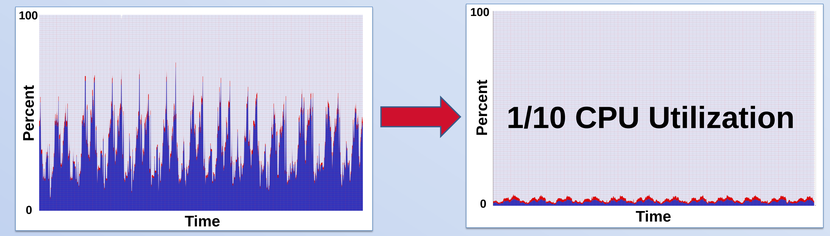


The DaNA approach uses the same GFP code as the OAS approach, with C-bindings for Fortran data and algorithm execution. This approach leverages the GFP use of NetCDF internally. OpenDAP is used to provide inputs, and the Java Native Interface (JNI) to the Data Fabric is used for outputs.





MEMORY / CPU FOOTPRINT



The Data Fabric holds, indexes, and provides all algorithm data: inputs, configuration, outputs. These diagrams show CPU Utilization after prototyping Data Fabric segmentation, where each segment holds a subset of data, organized by query name (e.g. starting with A-D in 1 segment, E-H in another, etc). The average decrease was by a factor of 10, a significant improvement. A Work Request is planned.

5 other options were considered but not analyzed further for this study.

CONCLUSION

The Algorithm Special Study provides techniques to reduce lifecycle costs that can be developed during PLT and promoted to the Ground System during the first round of algorithm updates. All L2 products require updates to match current AWG science algorithms. Two update tools: wrapping AWG algorithms, improving metadata generation.