ATBDs from the Implementer's Perspective

AWG Annual Review Meeting

June 7, 2010

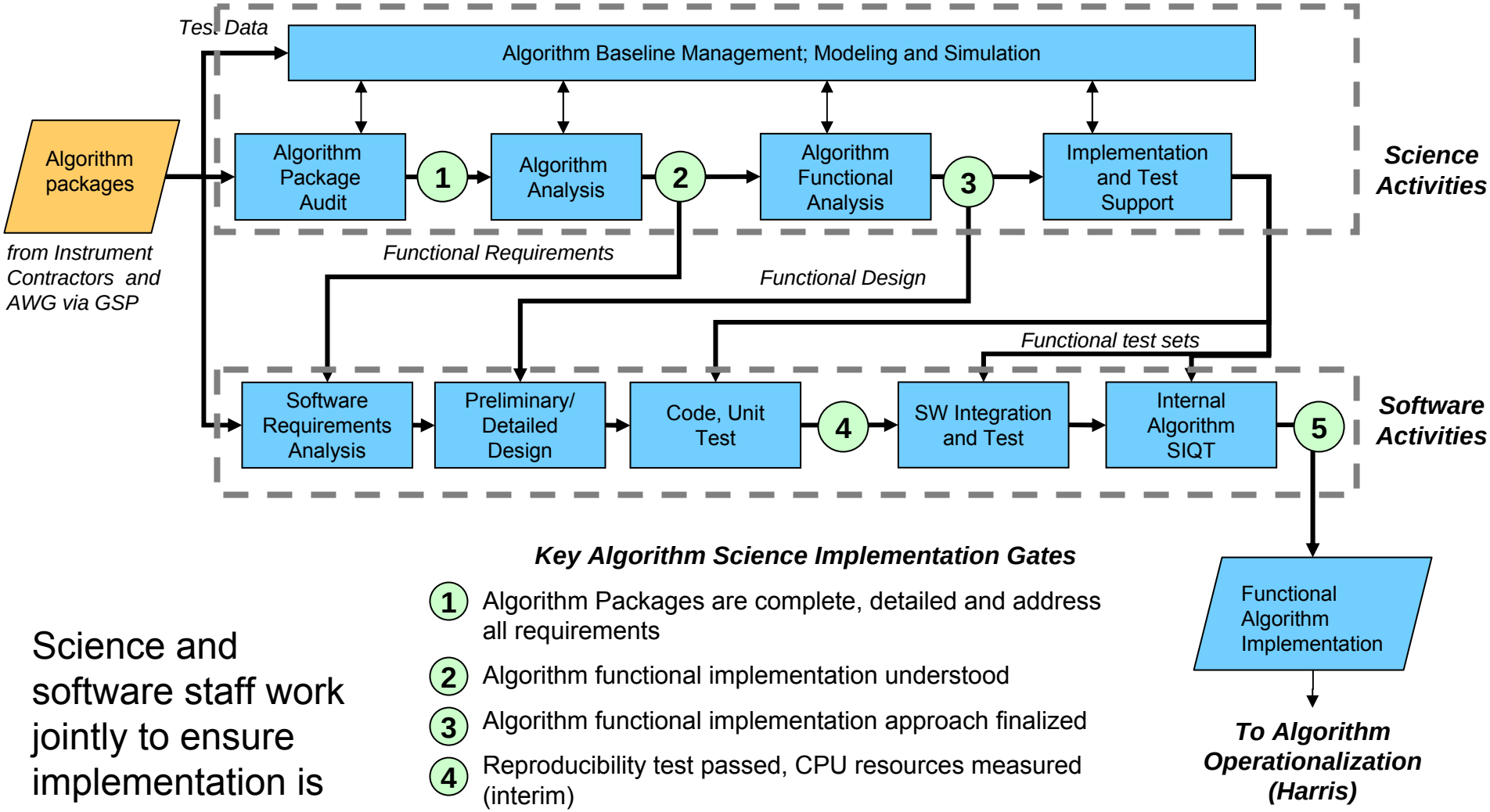
Ted Kennelly, Rich Lynch, Rob Braswell - AER

Rob Kaiser - Harris

L2 Product Generation Overview

- Core-GS Responsibility
  - To implement L2 algorithms defined in the ATBDs into operational code
  - To verify algorithm performance subject to the reproducibility requirement using AWG-provided test data to a very high level of agreement
    - Correlation: R-squared ≥ 0.9995
    - Outlier Restriction: No more than 1% of values differ by more than 0.15% (of the measurement range)
    - Classification Accuracy: ≥ 99% for categorical products
  - Requires detailed descriptions of all algorithm components
    - Inputs (including all required pre-processing)
    - Outputs (L2 products, Quality flags, metadata, etc)
    - All processing steps required to implement the algorithm
    - Clearly documented test data and configuration information

Algorithm Engineering Process

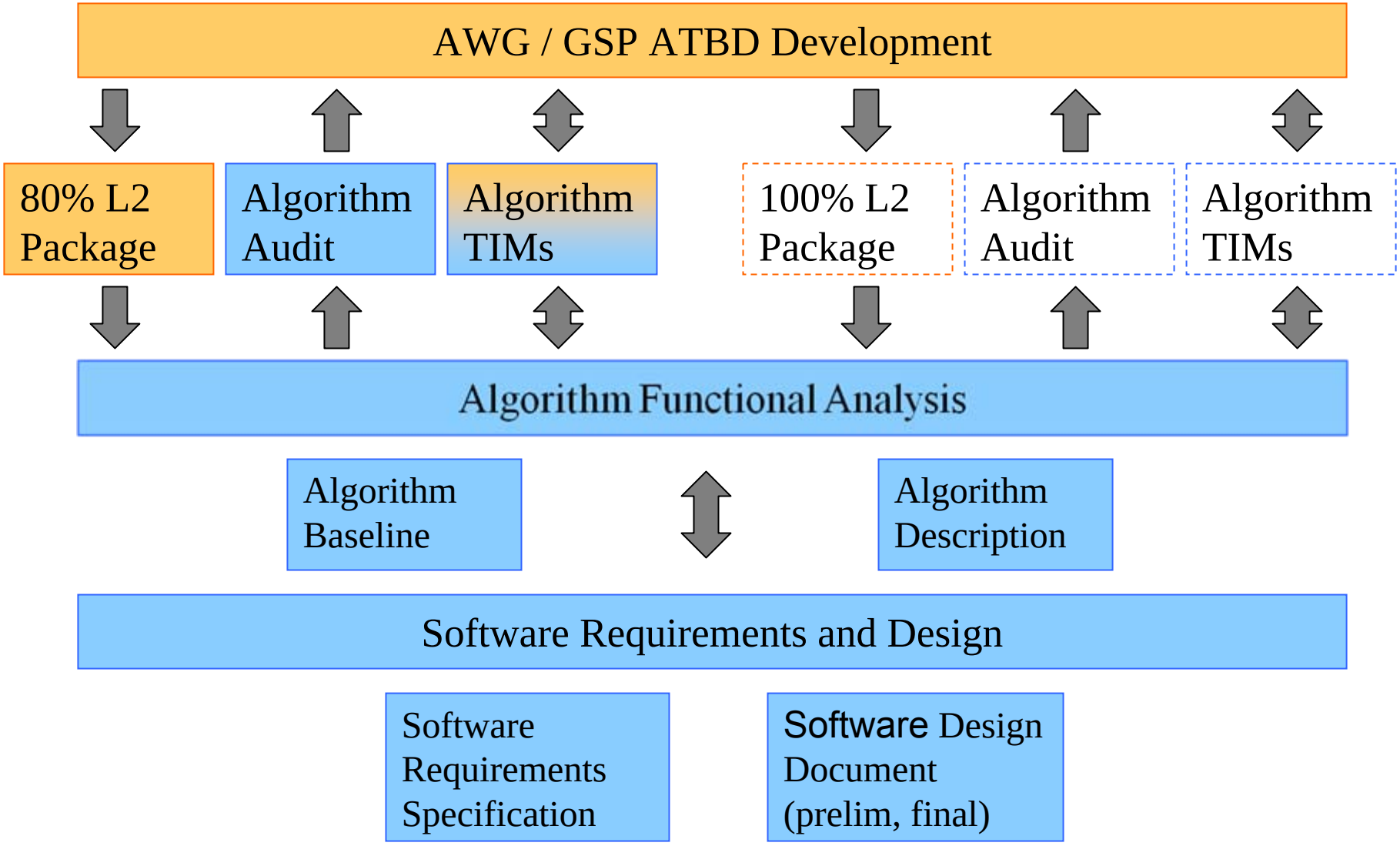


Key Algorithm Science Implementation Gates

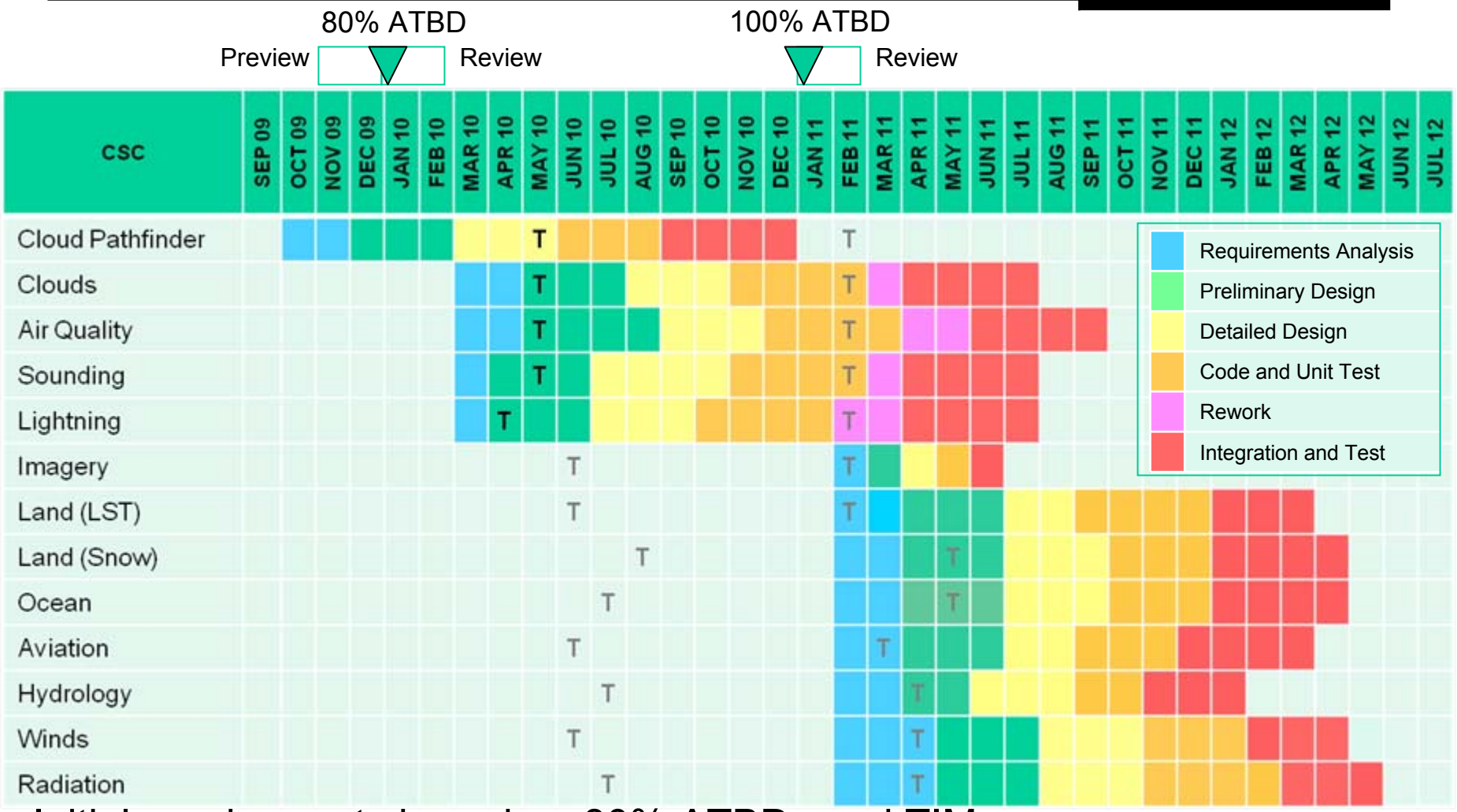
1. Algorithm Packages are complete, detailed and address all requirements
2. Algorithm functional implementation understood
3. Algorithm functional implementation approach finalized
4. Reproducibility test passed, CPU resources measured (interim)
5. Reproducibility and functional requirements verified, CPU resources measured

Science and software staff work jointly to ensure implementation is correct

Algorithm Analysis Interactions



Software Development Schedule: Chicklet Chart



- Initial requirements based on 80% ATBDs and TIMs
- Software already in Code & Unit Test when 100% ATBDs received

Under revision for ECP0004

Science/ Software Team POCs

Harris L2 Algorithm Lead: **Rob Kaiser**

AER Chief System Engineer: **Ted Kennelly**
AER Chief Scientist: **Rob Braswell**
AER L2 Algorithm Lead: **Rich Lynch**
AER L1b Algorithm Lead: **Christian Alcala**
AER Test Data Lead: **Yuguang He**

AER Chief Software Engineer: **Scott Zaccheo**

Algorithm	Science POC	Software POC
Cloud Mask	Jon Wrotny	Paul van Rompay
Cloud Type	Pam Puhl-Quinn	Keith Davis
Cloud Height	Bill Gallery	Greg Tully
Cloud Optical Depth (D)	Gary Meehan	Danielle Simmons
Cloud Optical Depth (N)	Jon Wrotny	Danielle Simmons
Aerosol Detection	Rich Lynch	Tom Calderwood
Aerosol Optical Depth	Bob d'Entremont	Owein Reese
Lightning Detection	Lizzie Lundgren	Dan Hunt
Sounding	Peter Finocchio	Cheryl Caldwell

Algorithm	Science POC	Software POC
Imagery	Gary Meehan	TBD
Land Surface Temp.	Lizzie Lundgren	TBD
Snow Cover	Rob Braswell	TBD
Sea Surface Temp.	Bill Gallery	TBD
Derived Motion Winds	Peter Finocchio	TBD
Hurricane Intensity	Peter Finocchio	TBD
Volcanic Ash	Rich Lynch	TBD
Rain Rate	Christian Alcala	TBD
SW Radiation	Gary Meehan	TBD
Fire	Rob Braswell	TBD

L2 Algorithm Package Audits

- Opportunities to provide feedback and resolve questions through interactions with the AWG / GSP is an integral part of our process
- Algorithm Package Audits were performed on all information provided in the 80% delivery and a summary report was released in February 2010
  - Evaluation of utility of L2 algorithm packages
  - ATBD content and maturity
  - Test data and documentation evaluation
  - Detailed questions on algorithm requirements and function
- Proceeding through software requirements and design, additional questions have been identified as we dig deeper into the algorithms
- Tools used to support this process
  - Issue database used to document questions and track status
  - Technical Interchange Meetings and resulting Actions
  - Algorithm Baseline & Algorithm Description Documents

L2 ATBD Review Criteria

Category	ATBD Assessment Criteria
Requirements	ATBD should state the product requirements and relate algorithm functionality to these requirements including identification of products, description of algorithm coverage, horizontal resolution, refresh, etc and including a discussion of requirements imposed on other algorithms introduced through dependencies.
Inputs	ATBD should clearly and unambiguously define and describe all the inputs required by the algorithm, these include: all sensor derived data, all ancillary data, all control parameters and databases. Any pre-processing steps must be fully described.
Outputs	All algorithm outputs need to be clearly and unambiguously identified and classified, i.e. as delivered products, intermediate product, diagnostic product, or quality flag. The intended use for all outputs and the method for calculation should be clear. All product metadata and methods of calculation should be described.
Theoretical Basis & Math Description	The ATBD should provide a theoretical basis section describing the physics behind the algorithm, along with mathematical representations of the key operations/processes.
Algorithm Logic	The ATBD should contain detailed descriptions of all the processing steps required to implement the algorithm. This includes: the operations involved, the data requirements for each step, the output for each operation, and conditions for termination.
External Routines & Databases	ATBD should describe all external routines (e.g., CRTM) and databases (e.g., LUT) and the mechanism for incorporating them into the algorithm flow. Information should also be provided describing the conditions/mechanism for which the tables will be updated/maintained.
Software Implementation/ Exceptions	The ATBD should describe any tuning required to meet/maintain product performance. All control variables should be clearly defined. If any operations could result in undefined results, i.e. divide by zero; conditions for avoidance should be clearly defined. Procedures should be provided to test for and update retrieved variables that are unphysical.

Looking Ahead to 100%

- The 100% Algorithm Package will be the final deliverable
- For algorithms already in CUT, our schedule requires a mechanism to achieve rapid closure on questions from this material
- We propose a multi-day TIM to address questions and to review the Algorithm Description Documents (or Software Design Document) with the AWG developers
- For algorithms just beginning implementation, we propose a series of TIMs to address questions in support software requirements analysis and design activities

Good Communication is the Key to Success

- Algorithm Development process is designed to provide visibility and good two-way communication with the algorithm developers
- Technical Interchange Meetings have been a highly effective mechanism for communication
  - Present state of our understanding of the Algorithms
  - Discuss components/processing steps that require clarification
  - Provide feedback to AWG on Algorithm Packages

TIM Schedule for 80% Algorithm Package Release

Product Group	Status	Product Group	Status
Lightning	approved	Volcanic Ash	6/11
Framework	approved	Winds	6/24
Clouds	approved	Rain Rate	7/8
Air Quality	approved	Radiation Budget	7/16
Sounding	complete	SST	7/22
Land	6/10	Snow cover	8/11
Imagery	6/11		

Life in the GOES-R Development Environment

7 June 2010

Robert Kaiser

Harris Corp

Introduction

- Objective of Development Environment (DE)
  - support L1b Calibration and Validation (Cal/Val)
  - support L1b algorithm maintenance
  - support L2+ algorithm/parameter maintenance
  - support software development for all GS elements
- Locations: Wallops & NSOF
- Information in this briefing was pulled from GS requirements documents, the OPSCON, or SDR material.
- Goal of this briefing is to start a dialog through the GSP facilitating the DE design.

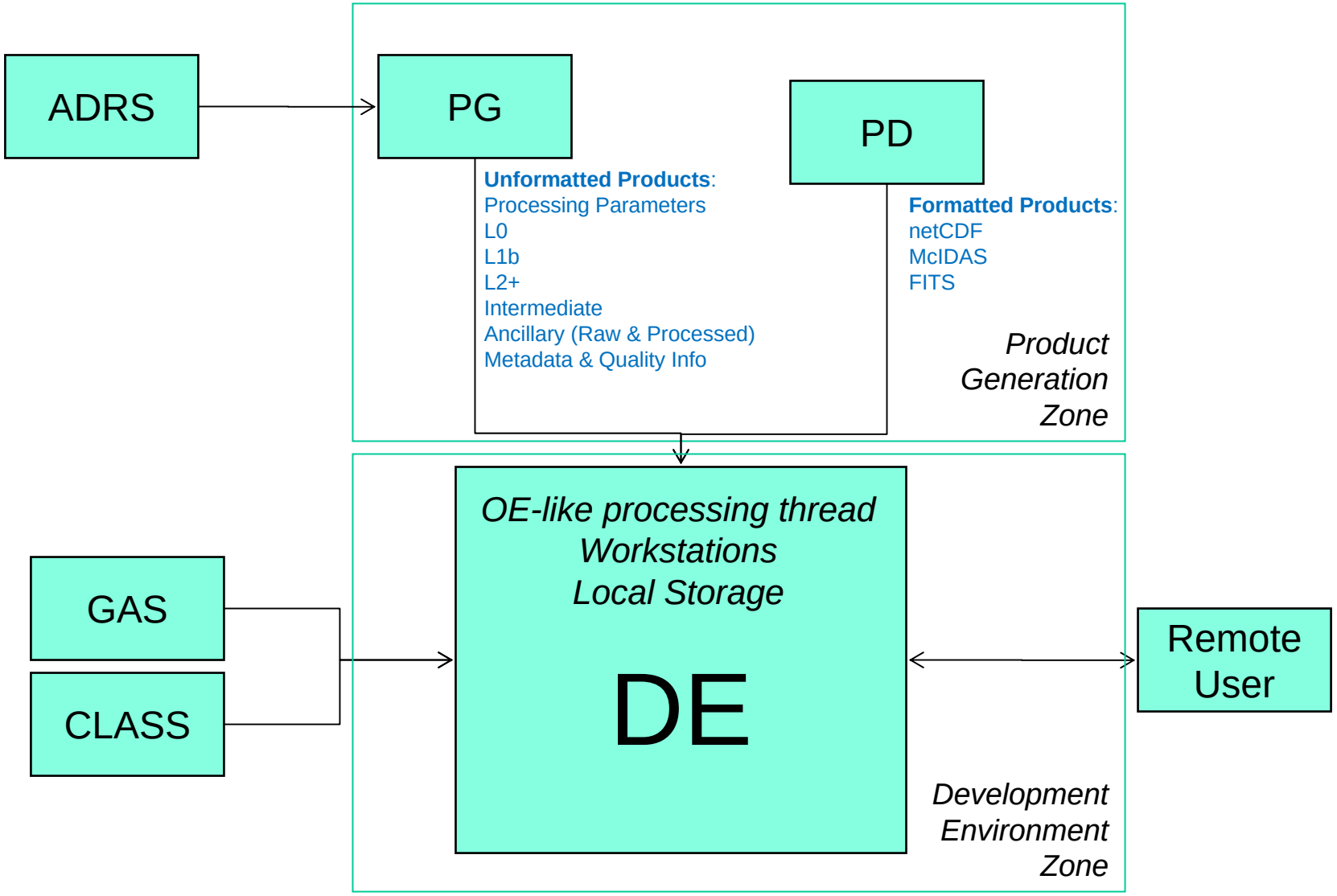
Hardware and Software

- Within the DE there will be
  - HW sized to process the data from one satellite
    - Servers w/local cache, local DDF, local storage (25 TB)
  - Workstations for Cal/Val ops, algorithm and SW maintenance.

Site	Subsite	Baseline	HW Function	Server Type	Qty
NSOF	DE	Baseline	Ancillary Data Ingest	Blade D	1
			Cache	Blade B	11
			Cache Staging	Blade E	2
			CAL/VAL	Mid-range Server C	3
			DB	Mid-range Server D	2
			Development Tools	Mid-range Server B	4
			Distribution Management	Blade C	2
			Enterprise Supervision	Mid-range Server A	7
			ESB	Mid-range Server A	2
			GRB Ingest	Blade C	2
			Image Sectorization and Storage	Blade E	2
			NAS	Mid-range Server D	2
			Product Distribution	Blade C	2
			Product Formatting and Storage	Blade E	2
			Product Monitor	Mid-range Server A	4
			Product Performance	Blade D	2
			Product Processing	Blade A	25
			Proxy Server	Mid-range Server D	2
			Remote Access	Mid-range Server A	2
			Removable Media	Mid-range Server C	1
			Resource Management	Blade C	1
			Satellite Operations	Mid-range Server A	2
			Simulators	Laptop	2
					3
			Standard Workstation	Workstation A	5
			Support Services	Mid-range Server A	5
			Workload Management	Blade C	2

Acrobat Reader	JTest	Patchlink Security Config	Tivoli Storage Manager for
Active VirusScan	JUnit	Management	Database (Linux)
ActiveTCL	Kakadu	Patchlink Security	Tivoli Storage Productivity
Ant	LAPACK++	Management Console Ent.	Center Standard Edition
Args4j	Linux Advanced Platform	Patchlink Update	TotalView Debugger
CDT	Linux HPC Compute	Patchlink Update Server	Understand (C++/Fortran)
Code Sonar	Linux Workstation	Perl	VNC
CodeCollaborator	LinuxShield	Photran	VTune Analyzer
Commons	Log4cxx	Policy Auditor Agent	Valgrind
Crystal Reports	Log4j	Python	Weblogic Suite
DRMAA	Mapping Toolbox	Rapid SQL Pro	Xerces XML Library
Doxygen	MatLab	Real Time Logic Pragmatic	enscript
ENVI+IDL	McIDAS-X	General Multicast	gMake
EPIC	NFS	Remediation Manager Agent	gdb
Eclipse	NetBeans	Remediation Manager Server	xCAT
Enerjy	NetCDF	SQLDeveloper	
FindBugs	Oracle Database Enterprise	Sanctuary Application Control	
Firefox	Edition	Sanctuary Application Control	
GCC	Oracle Diagnostic Pack	Server	
GMT Generic Mapping Tools	Oracle Provisioning Pack	Shuttle Radar Topography	
GPFS	Oracle Real Application	Mission	
Grand Total	Cluster (RAC)	Signal Processing Toolbox	
Hibernate	Oracle Servce Registry	Simulink	
Hudson	Oracle Service Bus	The Count	
Image Processing Toolbox	Oracle Tuning Pack	Tivoli Capacity Process	
J2EE SDK	PGI Server Complete	Manager	
J2SE	PHP	Tivoli Composite Application	
JAutoDoc	PMD	Manager for J2EE	
JDT	PTP	Tivoli Storage Manager	
JGraph	Patchlink Scan		

Connectivity, Storage, and Security Zones



Functions & Characteristics

- Read access / ingest
  - All inputs and outputs to individual and groups of algorithms in the OE. (L0, L1b, L2+, Auxiliary, Ancillary, Intermediate products (incl diagnostic)) – within 60 seconds of production
  - Data from OE persistent storage (2-day, life of mission, 5-day….)
  - CM Access (source code, LUTs, processing configuration parameters)
  - Reports (performance data, anomalies)
  - Proxy data
- Modify locally
  - Source code, control parameters, calibration parameters, alternate ancillary data
- Generate, display, capture, store, and export
  - Dynamic calibration parameters (ABI L1b rad/cal..)
  - Products from algorithms in groups or individually
  - Reports
- Misc
  - Compare versions of cal parameters, trend, and write reports on.
  - Document results
  - Cannot interfere with OE or ITE
  - Host algorithm maintenance and Cal/Val tools provided GFE
  - Generate and read GRB

Remote User

- The GS Development Environment shall provide
  - security-compliant access to the development environment
  - From external algorithm maintenance and Cal/Val facilities
  - without the need to run local software applications at, or transfer applications to, the external facilities

"Promotion" Process

- New algorithms are developed and tested in the DE.
- When they have completed testing, they are deployed in the ITE and tested using live input data and simulated external systems.
  - This allows new algorithms to be tested as needed using live data from the instruments.
- When this testing is completed, the algorithm software is deployed in the OE as an operational algorithm service.
- The SBA is configured using interactive software to add the new algorithm into the operational precedence chain.
  - The live input data is connected to the algorithm to process.
  - The outputs of the new algorithm are routed to the DE for evaluation. This configuration has no effect on the operational algorithms or the delivery of products.
  - Availability servers are used to host the new algorithm so that no latency impact occurs to operational products.
- When the new algorithm has been thoroughly evaluated, it is configured as the operational algorithm by connecting its products to the other operational algorithms and to the external interfaces.
- The old algorithm is configured to no longer run or to run in non-operational mode by routing its products to the DE.

Details to Work Out

- Need use cases for key functions
  - Establishing processing threads w/alternative control parameters or algorithms
  - Configuring a monitoring process
  - Managing multiple processing chains
  - Accessing data from GAS and CLASS
  - Batch processing data from GAS and CLASS
  - Exporting data
  - Remote access