JOHNS HOPKINS
APPLIED PHYSICS LABORATORY

An Extensible Architecture for Data Product Generation

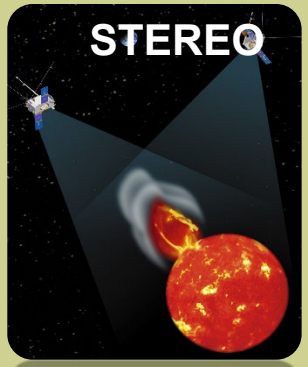
Adam Byerly
Senior Professional Staff
Johns Hopkins University
Applied Physics Laboratory
Adam.Byerly@jhuapl.edu

Agenda

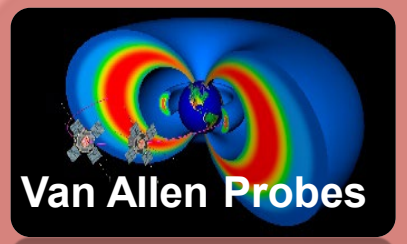
- APL Ground System Context
- What is a Data Product?
- How are Data Products generated?
- How are Data Product Generators updated?
- Moving towards a common architecture

Evolution of APL Ground Systems

1990s
One-Off Systems

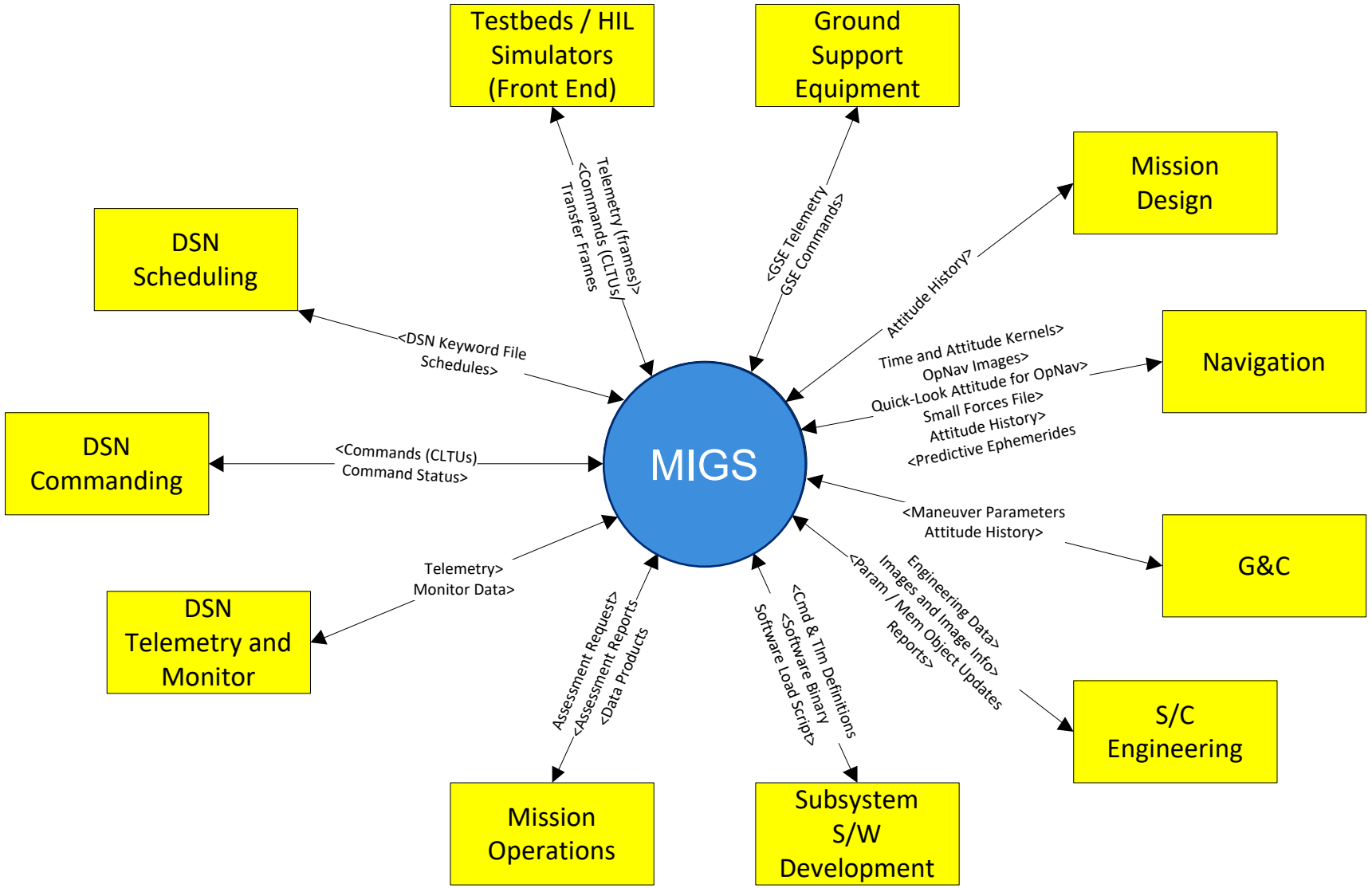


2000s
Common Ground Architecture



2010s – 2020s
Mission / System Independent Architecture

Mission-Independent Ground System (MIGS) Context



Vocabulary

- **Report** – any formatted document containing mission information (e.g. telemetry)
- **Data Product (DP)** – a report that conforms to specified format, location, and generation methods.
  - All Data Products are Reports, but not all Reports are Data Products
- **Data Product Generator (DPG)** – a software component that generates a specified DP
- **Report Generator** – a software component that generates a report
  - All DP Generators are Report Generators, but not all Report Generators are DP Generators
  - Note: “Report Generator” is being used as a generic term. It is NOT to be confused with specific software instances with similar-sounding names (e.g. report_generator, ReportGenService)

Example Data Products

- **Attitude History File**
  - SPICE C-Kernel
  - Contains spacecraft (and/or actuator) attitude information
- **Command History Report File**
  - ASCII Text File
  - Contains command status packet data
- **Time History File**
  - ASCII Text File
  - Contains correlated timekeeping data

Data Product Generators

- **Automated Software** – A software component runs automatically, based on time/event triggers, to generate the DP,
  - Example: Cron job to run hourly/daily/weekly

- **Manual Software** – A software component is run by a user to generate the DP
  - Example: User runs software to generate attitude history file for a supplied time range

- **Manually Generated** – A user generates the DP via some external method
  - Example: User extracts telemetry / data into an excel worksheet

Updating DPGs

- Not all missions are the same!

- While we can carry forward ”mission-independent” software between missions, “mission-specific” logic requires updates
  - E.g. Change to number / nature of actuators requires update to our attitude history file

- By minimizing / generifying our “mission-specific” logic, we reduce the size / frequency of updates

DPG Considerations

- **Data In**
  - How are we getting data in?
  - What is the format?

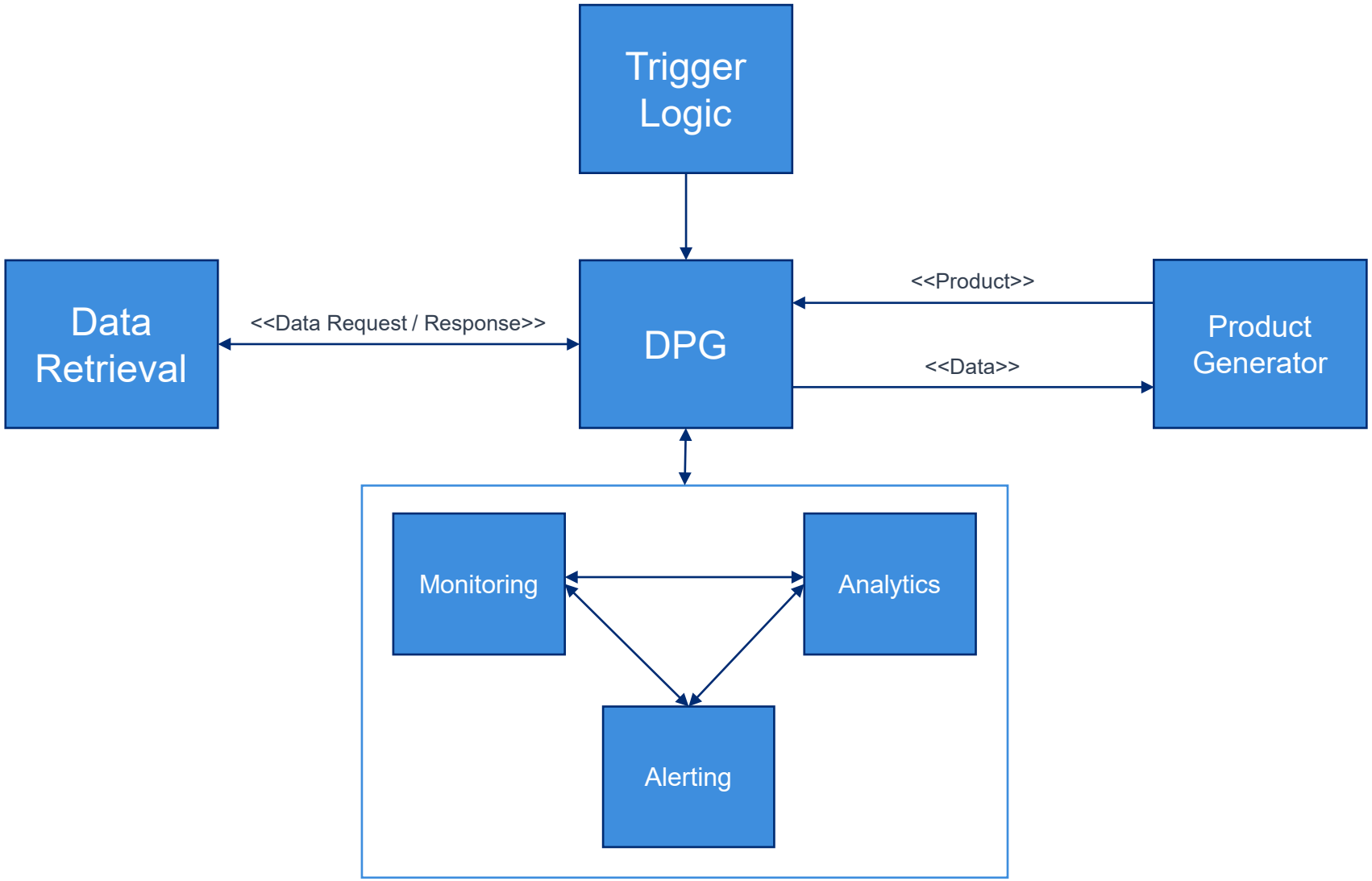
- **Data Out**
  - How are we getting data out?
  - What is the format?

- **Processing**
  - What processing is required to transform “data-in” to “data-out”?

- **Reporting / Alerting**
  - How do we alert the end-user a new product is available?
  - How do we alert the end-user an error / anomaly occurred?

- **Triggering**
  - When should we run?

Common DPG Framework



Example and Lessons Learned

- **Attitude History File**
  - SPICE C-Kernel
  - Contains spacecraft (and/or actuator) attitude information
  - Initially written for mission with:
    - Three actuator components: two solar arrays, one antenna
    - Five gimbal angles: two per solar array, one per antenna
  - Used legacy telemetry retrieval mechanism

- **Rewrite**
  - Defined initial interface abstractions and implementations
  - Used service-loader infrastructure
  - Existing mission migrated to new approach; step towards deprecating legacy retrieval mechanism

- **Main Lesson Learned**
  - Separation of concerns along service-lines allowed for easy adoption of new functionality across missions

JOHNS HOPKINS

APPLIED PHYSICS LABORATORY