Data Processing Pipeline With Transaction-Oriented Data Sharing

Thomas Huang
{Thomas.Huang@jpl.nasa.gov}

Larry Preheim
{Larry.Preheim@jpl.nasa.gov}

Jet Propulsion Laboratory, NASA
4800 Oak Grove Drive, Pasadena, CA 91109, U.S.A.

Abstract

Space science data processing involves executing a sequence of computing-intensive modules in an effort to produce high-quality science data products to further our understanding of our universe. *While the fundamentals have not changed, the process has certainly* evolved to take advantage of modern distributed computing and high-speed networking technologies. This evolution has enabled us to process science data at an extremely high rate and share processed results with our science communities in near real time. We often refer to the notion of data sharing as the ability to publish data on a well-known location where it is visible to others. However, higher quality-of-service (QoS) is expected when it comes to valuable science data.

This paper makes three contributions to the area of modern science data processing systems. First, the paper describes the science data processing pipeline, developed at the Multi-mission Image Processing Lab (MIPL) of JPL, for transforming raw space data into high quality image data and automating the distribution of data using a high-performance file transaction service. File Exchange Interface (FEI) is the file transaction service developed at MIPL. Second, it presents the FEI component architecture [4] in the area of file transaction management, security, and file integrity verification. Finally, the paper presents the federated model for the FEI service to demonstrate how to create a pool of file transaction services to support load balancing and service failover, and simplify service management.

1. Introduction

The recent Mars Exploration Rover (MER) mission has demonstrated that it is possible to produce high-resolution science images and deliver them to scientists in near real time. A quote from a recent article of The New York Times said, "The portal to Mars at marsrovers.nasa.gov/home includes nearly 1,000 Web pages filled with computer animations, panoramic and catalog of other images, some in 3-D, published online almost as soon as the scientists receive them [3]." One of the many innovations of the MER science data processing architecture is its ability to process raw science data and have them delivered in a completely lights-out fashion. The architecture involves a federation of intelligent systems that is able to query, process, catalog, and deliver data products in an efficient, reliable, and secure manner.

The fundamentals of science data processing have not changed. What has changed is the expectation of a shorter development cycle, tighten requirements on the quality-of-service, and technology that can be applied to take advantage of modern parallel computing and high-speed network communications. The science data processing pipeline developed at MIPL is designed as a reusable architecture to automate the process of raw science data and deliver final data products to subscribed parties.

2. Data Processing Pipeline

The MIPL data processing pipeline consists of a mission-specific telemetry processor, a collection of science processing modules, and a file registration and distribution service. The mission-specific telemetry processor is responsible for acquiring and process raw science data. It involves data decompression, reconciliation, and geometric correction. Data products are being produced and cataloged by the telemetry processor. Data products can consist of images, metadata, and non-image binary data. The collection of science modules includes modules for creating image mosaics, map projections, pattern-recognition, limb detection, and so forth. The inputs and outputs from these modules could consist of images, metadata, and non-image binary data.

2.1 The Traditional Approach

The traditional approach in science data processing requires analysts to manually invoke commands to begin the mission-specific telemetry processor followed by manually invoking the associated science processing modules. Science operation users will be notified via email with the physical location of the processed science data products. The science operation users are required to manually login to initiate file transfer to have the products delivered to their local system. After acquiring the science products, the science operation users may manually perform operations such as moving the acquired products into their local shared storage or invoke local processing modules on the acquired products.

It is obvious that the traditional approach described above requires a lot of user interactions, which is a performance bottleneck and expensive. The traditional approach addressed security with email, user logins and user groups. A high-profile mission such as MER requires hundreds of concurrent user logins. Some users are MER scientists and some are ordinary citizens. With the traditional approach, it would raise many *legal and additional security concerns. Another issue the traditional* approach failed to address is the secure file transfer to prevent premature release. Concurrency is another weakness of the traditional approach. It has no method of preventing users from getting a partially written file and it has no method of preventing a file from being updated or deleted while it is still being sent to other users. Finally, this traditional approach also failed to guarantee the integrity of the files being transferred.

2.1 The Pipeline Approach

What is missing from the traditional approach is a service that plays the roles of a security manager, a traffic cop, a messenger, and an insurance agent. As a security manager, the service addresses issues regarding user authentication, authorization, accounting, and communication. As a traffic cop, the service addresses issues *regarding concurrent accesses and modifications to data products.* As a messenger, the service addresses issues regarding automatic file delivery and event notification to enable automatic dispatch of local business logics. As an insurance agent, the service addresses the integrity of the data products it manages and delivers.

The MIPL data processing pipeline, in Figure 1, is centered on such file registration and distribution service, the File Exchange Interface (FEI). FEI organizes files by user-defined file types. Mission-specific telemetry processor is subscribed to telemetry data file types to be triggered as soon as Ground Data Service (GDS) publishes new telemetry data files to FEI. The collection of science processing modules are subscribed to the file types that consists of products generated by the telemetry processor to be triggered as soon as new products are available. The modules publish their processed results through FEI, perhaps with different file types. Science operation users, both locally and remotely, can subscribe to file types that consist of telemetry processor and science module outputs. In the case of the Mars Rovers' portal site, the delivery of a new data product also triggers local processing scripts to create new formatted HTML pages for public viewing.

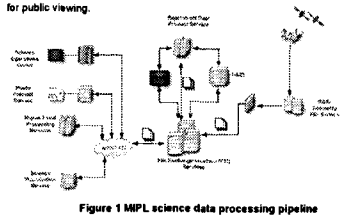

Figure 1 MIPL science data processing pipeline

3. File Transaction Service

A transaction in database systems is a logical unit of work that possesses the properties of atomicity, correctness, isolation, and durability ACID [2]. In atomicity, the sequence of operations within a transaction must all execute with correct results or a rollback must occur to restore the database to a state prior to the transaction. Correctness of the data refers to its consistency and its integrity. A transaction must be performed in isolation to avoid other processes from accessing data that is involved in the transaction. After a transaction is committed, its result must be available and persisted without ambiguity.

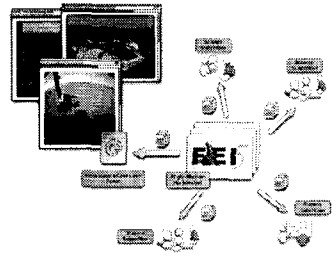

Figure 2 FEI as a product delivery service

FEI is a generic data product distribution system, in Figure 2, and registration service that is integrated as a service component of the science data processing pipeline. The service organizes data products through user defined file types and offers secure, high-performance file transaction and distribution capabilities that conventional file distribution service lacks. A file type in FEI is users' interface to query, subscription, registration, and retrieval files of a specific type.

3.1. Component Architecture

FEI is designed and developed as a component-base framework that depends on a set of virtual components [1] to enable pluggable configuration of mission specific business requirements. The core service components, in Figure 3, of FEI consist of file service configurator, file transaction manager, session manager, protocol pipeline, subscription manager, and monitor service. The deployment diagram also identified a few virtual components, depicted with a circle followed by a solid line attached to a physical component. A virtual component consists of an abstract interface that is required by the core framework. The actual implementation of the component to be used will be determined during deployment time. The pluggable component framework offers system architects the options to select the service components that are a best fit to the mission requirements.

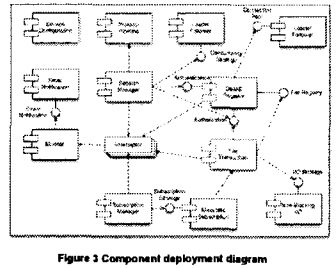

Figure 3 Component deployment diagram

3.2. File Transactions

FEI could be viewed as a database of files that supports file queries, registration, and acquisition. All transactions performed on FEI possess the ACID properties, which prevents cataloging of any incomplete files and access to files and data that are in transaction, in Figure 4. Files that are in the process of being registered to FEI are not visible to other concurrent users until the registration is complete. A file that is in the process of being delivered to interested parties is prevented from being modified or removal until the transfer is complete.

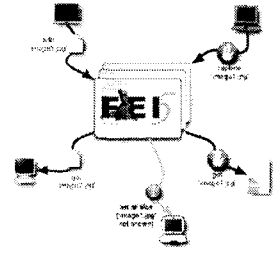

Figure 4 Transactions in FEI

3.3. File Integrity

As a file service, the integrity of the files it manages is vital. Within the context of FEI, file integrity enforcement is not limited to the file system where upon which the client/server applications are residing. It must be able to detect, notify, and recover from any corruptions occurred during transferring of files. Both client and server processes compute checksums during file transfer and compared upon completion of the transfer for corruption detection, in Figure 5.

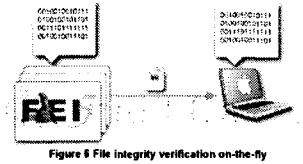

Figure 5 File integrity verification on-the-fly

3.4. File Subscription

One of the many strengths of FEI is its support for data subscription, in Figures 6. This allows users to register requests to have new files delivered as soon as they are available to the FEI server. The subscription client can register a local process to be invoked when a new file is delivered. As a reactive [5] agent, the client can now register local business logic to react as soon a new data product has arrived.

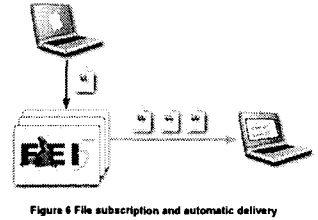

Figure 6 File subscription and automatic delivery

4. Service Federation

High-profile missions such as MER always attract large number of users. A scalable data distribution service must be to handle large number of concurrent requests. In the case of a file transaction service, this also means to be able to handle large number of concurrent transactions as well. Since there are physical connection limits to any single host machine, the solution that FEI offers is to have a collection of associated FEI servers. File transactions performed by each FEI server need to reflected to and visible by other FEI servers within the connection. This is the durable property of ACID. A federation model, in Figure 7, is used to utilize the service registry as the transaction communication channel among the servers.

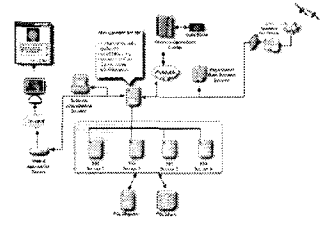

Figure 7 FEI federation model

The FEI federation model is also equipped with a management service that is provided as a service registry and a load balancer. As a service registry, the management service manages a file type lookup table with the list of associated connection information. As a load balancer, the management service actively gathers performance statistics on all the servers it manages. Upon receiving a client connection requests, the management service lookups up and *returns a server reference with the least load. This model allows the* FEI administrator to add new servers into the federation without having to distribute new configuration information to the FEI users.

5. Conclusion

Using a file transaction service as the secure messenger among service components has been proven to be a reliable approach in implementing any automated data processing architectures. The service decouples service components from one another by offering service components to communicate through user-defined file types. Each service component is only requires to implement its domain-specific logics on how to react upon receiving new data products and may choose to share its processed results with others with well-defined interfaces. New service components can be added to the pipeline without affecting other existing components.

6. Reference

[1] A. Corsaro, D. C. Schmidt, R. Klefstad, and C. O'Ryan, "Virtual component: A Design Pattern for Memory-constrained Embedded Applications," Proceedings of the 9th Annual conference on the Pattern Languages of Programs, Sept. 2002.
[2] C. J. Date, An Introduction to Database Systems, Eighth Edition, Pearson Education, Inc. 2004.
[3] A. Harmon, "Eager NASA Is Bring Mars Down to Earth." The New York Times, January 27, 2004.
[4] T. Huang, "Component Architecture – The Software Architecture for Mission Requirements." In proceedings of The Fifth International Symposium on Reducing Cost of Spacecraft Ground Systems and Operations (RCSGSO), Pasadena, California, USA, July 2003.
[5] D. Schmidt, M. Stal, H. Rohnert, and F. Bushmann, Pattern-Oriented Software Architecture, Volume 2, Wiley & Sons, New York, 2002.