The Design of a Petabyte Archive and Distribution System for the NASA ECS Project

Parris M. Caulk
LORAL Aerosys
1616A McCormick Drive
Landover, MD 20785
pcaulk@eos.hitc.com
Tel: +1-301-925-0610
Fax: +1-301-925-0327

Introduction

The NASA EOS Data and Information System (EOSDIS) Core System (ECS) will contain one of the largest data management systems ever built - the ECS Science and Data Processing System (SDPS). SDPS is designed to support long term Global Change Research by acquiring, producing, and storing earth science data, and by providing efficient means for accessing and manipulating that data. The first two releases of SDPS, Release A and Release B, will be operational in 1997 and 1998, respectively. Release B will be deployed at eight Distributed Active Archiving Centers (DAACs). Individual DAACs will archive different collections of earth science data, and will vary in archive capacity. The storage and management of these data collections is the responsibility of the SDPS Data Server subsystem. It is anticipated that by the year 2001, the Data Server subsystem at the Goddard DAAC must support a near-line data storage capacity of one petabyte (10^{15} bytes).

The development of SDPS is a system integration effort in which COTS products will be used in favor of custom components in every possible way. Some software and hardware capabilities required to meet ECS data volume and storage management requirements beyond 1999 are not yet supported by available COTS products. The ECS project will not undertake major custom development efforts to provide these capabilities. Instead, SDPS and its Data Server subsystem are designed to support initial implementations with current products, and provide an evolutionary framework that facilitates the introduction of advanced COTS products as they become available.

This paper provides a high-level description of the Data Server subsystem design from a COTS integration standpoint. It discusses the service-oriented basis for the design, how the design supports multiple COTS technologies, how the design supports scalability and the introduction of new COTS technologies, and how the design addresses the management of I/O between the archive and the SDPS Data Processing subsystem.

SDPS Overview

SDPS [1] will support the services required to ingest, process, archive, manage, and access science data and related information from the entire EOSDIS. A typical DAAC will consist of the following SDPS components :

- An Ingest subsystem for acquiring all data from EOS instruments, NASA Probe flight missions, and other remotely-sensed data.

- A Data Processing subsystem for the generation of science data products from ingested instrument data, and from previously stored data products. The Data Processing subsystem executes hundreds of science algorithms which generate hundreds of gigabytes of science data on a daily basis. The subsystem generates the bulk of the data stored into the archive, and accounts for a large portion of the retrieval demand for archived data.

- A Planning subsystem for planning the execution of data processing tasks.

- A Data Management subsystem that provides functions for locating and accessing data distributed among ECS DAACs and other data systems with which ECS interoperates.

- A Data Server subsystem with a capacity (at the largest DAACs) to archive, retrieve, and distribute hundreds of gigabytes per day.

- An Interoperability subsystem that advertises ECS data and services to ECS clients.

A Service-Oriented System

The design of SDPS is based on a service-oriented approach to data search and access. Predecessor systems have employed a "product-ordering" approach that derived from the file-oriented retrieval mechanisms provided by conventional hierarchical storage systems. The contemporary science user requires sophisticated search and access methods which can locate and manipulate required data using a variety of search and processing criteria. Normally, the science user will only be interested in a fragment of a data product. Subsetting and subsampling operations will be applied to ECS data after it is retrieved and before it is distributed to the user. Ideally, all of the data would be stored inside of a data base system that could provide advanced search, access, and data manipulation capabilities on a petabyte of data in tertiary storage.

The service-oriented approach adopted by SDPS is intended to support the eventual introduction of advanced mass-storage data base technologies when they become available. The approach has the following features:

- It provides the science user a set of services on earth science data, instead of providing copies of canned data products. The user issues service requests which typically involve the retrieval of stored data followed by one or more processing steps on that data. In effect, the user accesses data "objects" instead of data products.

- It associates data and services according to science "views" of the data; usually a given view corresponds to a science discipline (e,g. Atmospheric Dynamics).

- It provides a logical "Data Server" to provide each discipline-oriented view of the data. Each data server logically associates related data even though the data may be stored on a variety of physical devices and managed by different data management software. The science user accesses and manipulates the data as if it were stored and managed in the same manner.

- It supports the experimental development and introduction of advanced user interfaces and advanced data management methods by providing application program interfaces to supported services.

Data Server Subsystem Overview

The SDPS Data Server subsystem provides the resources and services required for the

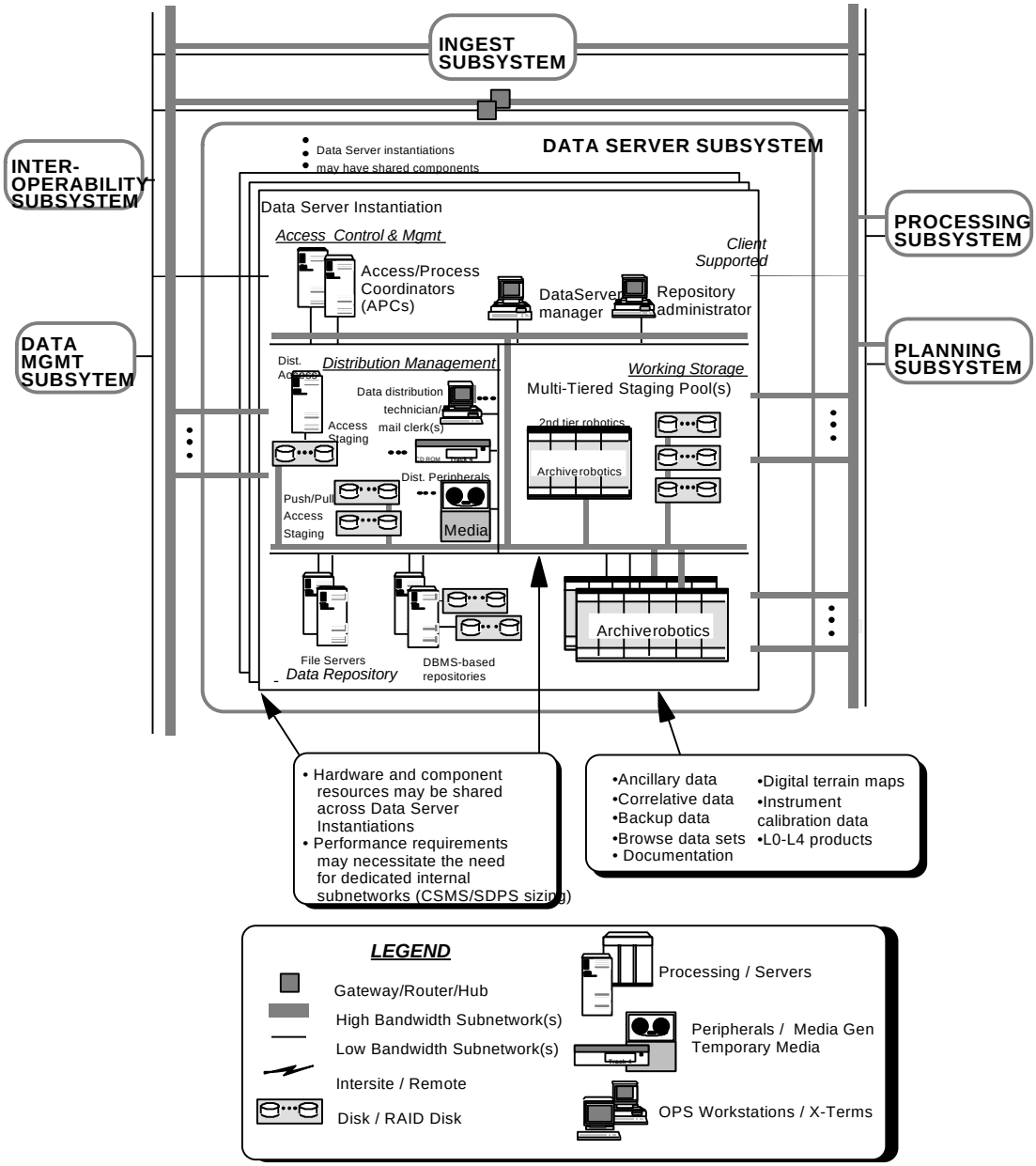

Figure 1 SDPS and the Data Server Subsystem

storage, management, and access of the ECS science data collections. Figure 1 illustrates the subsystem and its relationship to the other parts of SDPS. The Data Server

subsystem services data storage requests from the Ingest and Data Processing subsystems, and provides data search and retrieval services to ECS users and to the Data Processing subsystem. The subsystem manages the distribution of requested data to both hard media and to network clients.

Science users access the data and services provided by the Data Server subsystem by issuing service requests, either to the subsystem directly, or through the distributed access facilities of the Data Management subsystem. Users can request that processing (e.g. subsetting and subsampling) be performed on archived data products and have the result, or "result set", distributed to them. Additionally, the users can request processing on the result sets produced by previous service requests. The source data products for these operations can be many gigabytes in size. The capability to provide users the capability to perform processing at the archive site is essential because the size of the input data products is so large and the bandwidth of the distribution network is limited.

Physically, the Data Server subsystem is composed of the following major components:

- *Access components* - These support user/client access to data and services; they manage client sessions and manage the execution of service requests; and they advertise the services performed by the subsystem.

- *Data Repository* - This component provides storage servers for the storage, searching, and retrieving of data. A storage server can be any software/hardware subsystem that stores and retrieves data on demand.

- *Distribution* - This component supports the distribution of ECS data to users via networks and through the generation of hard media..

- *Working Storage* - This component manages the resources required for the temporary storage and buffering of ECS data.

The Access Component

The Access component supports client access to ECS data and services. It provides the interface between clients and the Data Repository, and is designed to support the use of COTS products in the implementation of the Data Repository. The client may be an interactive application system that is controlled directly by the user, an intermediary process provided by the SDPS Data Management subsystem, or a client belonging to the Data Processing subsystem. The client issues service requests for data and services on behalf of the user; the Access component manages the execution of the requests and directs them to appropriate Data Server or Data Processing subsystem components. Storage and retrieval requests are directed to storage servers (see Figure 2). Processing requests are executed directly by the Access component, or are directed to the Data Processing subsystem.

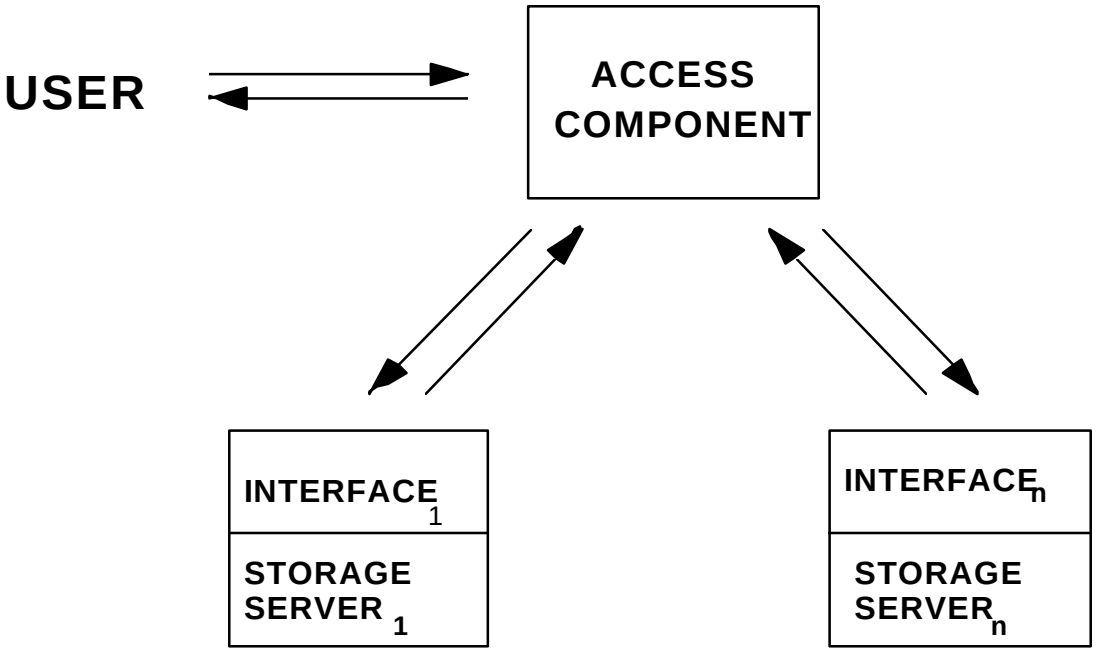

Figure 2. Access to Storage Servers

From the client perspective, the client is not interacting with the Access component or the storage servers, but is accessing a Data Server dedicated to a set of data and services that apply to a particular science discipline. The Access component can implement the logical functionality and services of multiple Data Servers. Data logically associated with a single Data Server may be physically stored across multiple storage servers. This feature of the design allows logically related data to be supported with heterogeneous storage technologies, and therefore supports system expansion and evolution.

A major function of the Access component is to insulate the client from the mechanisms used to store and retrieve data. In retrieval operations, the user specifies the data he wants and generates a service request containing a unique ECS Universal Reference (UR) to the data. The format of the UR is independent of the storage server and its underlying storage technology. The Access component uses the UR to determine which storage server contains the required data. The Access component then obtains the required data by directing the request, in a standard format, to the storage server.

Data Repository

The Data Repository component provides the storage servers for the storage, searching, and retrieving of data. It is designed to support the expansion of storage capacity and the introduction of new storage technologies using COTS products.

A storage server provides permanent data storage and retrieval services. It is composed of a hardware storage system and a COTS storage management software system. The storage management software manages the hardware and data, and processes service requests. Examples of storage servers include DBMS servers and tertiary storage systems managed by COTS File Storage Management products.

The vast majority of ECS data will be stored using technologies which support high storage densities at the lowest cost. At this writing the ECS project has not selected the storage systems for any of its releases. However, this paper assumes that one or more robotic tape libraries will be employed in the storage servers that store the bulk of ECS data.

The COTS software packages available for managing storage servers use different mechanisms for identifying and accessing data. Each storage server will have a custom software "wrapper" to translate standard service requests, from the Access component, into service requests that are compatible with the underlying COTS storage management software. Part of the translation process will necessitate the translation of the ECS UR into a product-dependent data set (or file) identifier. A storage server could be replaced with another one based on different COTS products without affecting the ECS data identification scheme, provided a translation interface is built for the new server.

The Access and Data Repository components support the use of multiple and heterogeneous storage servers. This design reduces dependence on a single storage technology. The design allows the use of a mix of storage technologies tailored to the variety of data that must be stored. It supports system evolution by permitting new technologies to be introduced and old ones replaced, in a gradual manner. Finally, it supports the expansion of the storage system through the addition of storage servers.

Working Storage

The Working Storage component provides high-performance disk storage, i.e. "working storage", for the temporary storage of ECS data. Specifically, it (1) stores files retrieved from Storage servers, (2) stores the result sets generated by the Access component and the Data Processing subsystem, and (3) buffers data to be inserted into the storage servers. Working storage roughly corresponds to the secondary storage component in a conventional hierarchical storage system.

One of the uses of working storage is to support interactive sessions with users. During these sessions, result sets are generated by the execution of service requests and are placed in working storage. The result sets may be browsed, processed further, or distributed as directed by a subsequent service request. The Access component must be able to retain result sets in working storage until the session with the user is terminated.

The Data Processing subsystem requires a large portion of working storage in order to support the execution of hundreds of science algorithms daily. The amount of working storage required to support the Data Processing subsystem is a major cost driver for the development of SDPS. The execution of a single science algorithm may process multiple gigabytes of input data contained in multiple files, and may generate many gigabytes of output data.

To a significant degree, the output of one algorithm is used as input for the execution of another algorithm within a few minutes or hours. Ultimately, most of the algorithm output will be archived. The retrieval load on the archive can be significantly reduced if algorithm output can be retained in working storage long enough to be used as input by algorithms that require that output. The amount of working storage required to support processing depends on the required retention intervals for algorithm outputs. Algorithms are scheduled in order to minimize these intervals. The efficient management of working storage requires that the Data Processing subsystem control working storage file retention in coordination with algorithm scheduling.

Mass-Storage I/O Management

The greatest design challenge for the Data Server subsystem is the management of the massive I/O (multiple terabytes per day) between the mass-storage library and the Data Processing subsystem. The Data Server design approaches this problem by supporting the scaling of I/O capacity and the intelligent management of working storage.

The conventional approach to managing multi-terabyte mass storage is to use a COTS File Storage Management System (FSMS) hosted on a single supercomputer. All I/O must pass through the FSMS host. Scalability is achieved under this approach in a "vertical" fashion by expanding the power of the host computer. This approach will not meet long-term ECS requirements for multi-terabyte daily I/O throughput rates. Ideally, I/O to and from the media drives in the mass-storage archive would be conducted from and to working storage, along parallel I/O paths which bypass the FSMS host. Current FSMS products do not support all of the capabilities required to implement this I/O architecture, nor do they meet ECS requirements for the application control of working storage. The ECS project is anticipating the development of FSMS products that will meet these requirements.

The ideal FSMS product would have the following major features:

- A volume management capability for controlling a variety of storage devices, including robotics.
- A file management capability that formats and organizes files on tertiary media.
- A capability to automatically monitor the integrity of the data in tertiary storage and monitor condition of the media.

- The capability to direct I/O between (1) the media drives and working storage and (2) between the media drives and a specified computer, without traversing the FSMS host.

- An application interface to working storage that allows the application to control the staging and retention of files in working storage.

The ECS near-term approach to scalability is to increase volume and throughput capacity "horizontally" by replicating storage servers supported by processors of moderate size. The current COTS-based design is intended to support the initial Releases of ECS and to support the introduction of high-performance I/O technologies later on.

Figure 3 illustrates a model for the initial implementation of a mass-storage storage

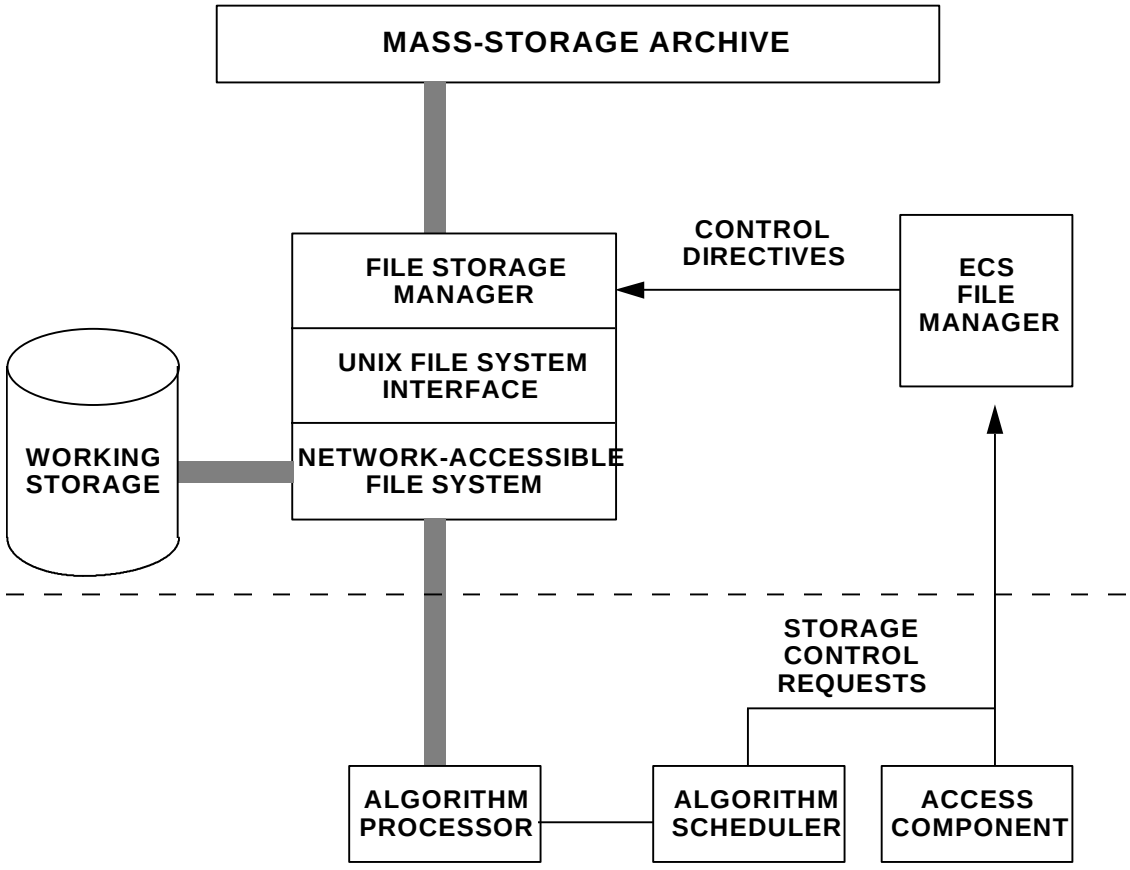

Figure 3. Initial I/O Management Approach

server that (1) could be implemented with available COTS products, (2) could be evolved to support advanced I/O and storage management capabilities, and (3) provides

for application control of working storage. In the model, a COTS FSMS is used to manage the files in the mass-storage archive, control the archive hardware, and move files between a mass-storage archive and working storage. Working storage is implemented by a disk array managed by a COTS network-accessible file system.

The FSMS and the network-accessible file system are separate products that interact across a standard UNIX file system interface. The separation of these capabilities is necessary in order to provide ECS flexibility in the selection and replacement of corresponding COTS technologies.

An essential feature of the model is that the management of files in working storage can be controlled by components external to the FSMS. This allows the Access component and the Data Processing subsystem to control the retention of files in working storage and allows the Data Processing subsystem to direct the staging of files prior to the execution of science algorithms. External components exercise this control through an "ECS file manager". The ECS file manager does the following:

- Accepts external requests for staging files, for retaining and releasing files from working storage, and for storing files in the archive.

- Keeps track of which files are in working storage and assigns a retention interval to each file.

- Directs the FSMS to migrate files between the archive and working storage, in response to external requests.

Conclusion

The Data Server subsystem design is constrained by the availability of applicable COTS technologies and the requirement to evolve to accommodate increasing system loads and advances in mass-storage technology. The efficient management of its storage resources require that the application have control over file caching mechanisms and over the movement of data between storage resources. Required support for multi-terabyte daily I/O rates necessitate the ultimate use of advanced I/O architectures. Currently no product (or family of products) provides a comprehensive approach to these issues. The current COTS-based design is intended to support the initial releases of ECS with a combination of COTS products, and will support the introduction of high-performance I/O technologies in later phases of the project.

Acknowledgments

This paper is based on the design efforts and helpful comments of Mark Huber, Tom Smith, and Alla Lake of LORAL Aerosys, and Eric Dodge and Evelyn Nakamura of Hughes Applied Information Systems.

References

1. EOSDIS Core System Project, "System Design Specification for the ECS Project" (194-207-SE1-001)