The Pleiades fractionated space system architecture and the future of national security space

David M. LoBosco* and Glen E. Cameron
Orbital Sciences Corporation, Dulles, VA 20166

Richard A. Golding and Theodore M. Wong
IBM Almaden Research Center, San Jose, CA 95120

Our paper is the first in a series of publications to document the development of a fractionated space system. We explain the derivation of the technical approach for system development and present the preliminary architecture. We will publish future papers following the PDR, CDR, and flight demonstration.

I. Introduction

In recent years, national security space systems have been plagued by cost overruns, schedule slips, and requirements creep. Notable programs have received much coverage in the press after incurring multiple Nunn-McCurdy cost growth breaches (total program cost increase by greater than 25%) or being cancelled outright. While these programs are the most visible, they are merely examples of the widespread problems with the government's and industry's approach to military space.

While a spacecraft that is never launched obviously provides a poor return on government investment, those spacecraft that have reached orbit may be equally bad investments if mission needs have changed or a critical component fails soon after launch. Recent studies[1,2] suggest that fractionated space systems can produce a higher value return on investment than traditional monolithic systems. Rather than continuing to design large monolithic satellites with unrealizable and competing requirements for performance and lifetime, we plan to develop a fractionated space system architecture, informed by value-centric engineering, that enables rapid initial operational capability through staged deployment, flexibility to changing national security needs, and robustness against attack and failures.

The Pleiades architecture implements fractionated space systems, where the system's functionality is spread over multiple, heterogeneous spacecraft modules. Each spacecraft module is a free-flying entity that has its own set of typical spacecraft bus functions but carries a unique mission payload or a resource such as a mission data processor, solid state recorder or high bandwidth downlink. The spacecraft modules fly in a cluster, and communicate with each other through a shared wireless network.

The Pleiades architecture allows for rapid response to operational needs, whether it be by launching new spacecraft modules to join a cluster or by retasking existing spacecraft resources. Modules carrying new payloads or resources are produced more quickly and at lower cost than traditional systems by using commercial space best practices, single string designs (system reliability is achieved through redundancy across spacecraft modules), and the benefits of economies of scale. This can lower the barrier to deploying space-based assets for agencies that cannot currently afford them. Also, the cluster is less vulnerable to attack, and can disperse itself when there is an incoming threat. When a failure does happen, the architecture provides for a smooth degradation in capability until a replacement can be launched and brought in to the cluster.

The fundamental innovation of this architecture is that information integration, not physical decomposition, is the key to realizing fractionation. In current space systems, the main inhibitor to rapid evolution and adaptation is stovepiped architectures that inhibit movement of information between elements, including the ground, and prevent reconfiguration of information flow to meet evolving mission needs. A fractionated space

*AIAA Member.

system architecture built on information integration can provide the space industry with orders-of-magnitude improvements in flexibility and robustness, just as the Internet and World Wide Web revolutionized ground-based commercial systems in the last two decades.

In this paper we present the preliminary results of phase I of the Pleiades project. We highlight the key features of the fractionated space system notional architecture and explain how it provides for rapid deployment, flexibility to changing mission needs, and improved space asset survivability. Additionally, we introduce our approach to information integration with the Virtual Mission Bus and describe preliminary approaches in networking and wireless communications.

The Pleiades project is part of the DARPA System F6 program. The System F6 program aims to demonstrate that a fractionated satellite architecture is feasible and useful for national security space missions. The program includes efforts in networking, wireless communication, distributed computing, wireless power transfer, cluster navigation, and distributed payload operations. It also includes efforts to build econometric models of the value that can be obtained from fractionated systems, to help guide design and acquisition in future systems.

II. Technical approach

The Pleiades project is structured in four phases. At the highest level, these phases follow the traditional phases of spacecraft design. Phase I consists of preliminary design culminating in a preliminary design review (PDR). Phase II consist of detailed design and culminates with a critical design review (CDR). Phase III is integration and test and Phase IV is launch and on-orbit operations. We focus in this paper on our work for phase I, which began in February 2008 and will be roughly halfway completed at the time of publication.

While the program phases are somewhat conventional, phase I includes a much larger scope than just spacecraft preliminary design. Additional tasks include six enabling technology studies, design and fabrication of a "hardware in the loop" testbed, and development of a value-centric design framework. The key technology areas are wireless communications, networking, wireless power transfer, cluster flight, distributed computing, and distributed payload operations.

During the proposal phase, we developed a notional system design, described in section III, that serves as a point of departure for the phase I effort. The program plan calls for staggered commencement of six fixed-duration technology studies followed by the preliminary design development. At this point, we have completed the wireless communications and networking studies, while we are in progress with the other four studies. In most cases, the outputs of the technology studies are a detailed trade space and a framework for value-centric design downselect. Once all of the studies are complete, we will evaluate the component technologies from a system-level point of view to maximize value.

III. Preliminary system architecture

We motivate our Pleiades system architecture by considering a customer with a notional Low-Earth-Orbit Earth science mission deploying three electro-optical (EO) imagers. Each imager has its own value, but the co-registration of these images enhances science value. One imager is more mature than the others; one is in the earliest stages of technology development; and one has a very short operational lifetime. In addition, the customer needs to store large amounts of data and perform image processing on board to produce extracted summary data products. The customer also desires the ability to send command uploads at any time of day or night to respond to unforeseen events such as seismic activity.

The notional Pleiades system contains five 225-kg (wet mass) modules and two 75-kg (wet mass) modules for a total of seven modules. The spacecraft are named after the nine brightest stars in Pleiades cluster: the Seven Sisters of Greek mythology—Asterope, Merope, Electra, Maia, Taygete, Celaeno, and Alcyone—and their parents—Atlas and Pleione. The 225-kg class module is named after Atlas; the 75-kg class module is named after Pleione. The five Atlas-class modules are named after the brighter stars, Alcyone, Electra, Maia, Merope, and Taygete. The two Pleione-class modules are named after the dimmer stars, Celaeno and Asterope. An Atlas-class and Pleione-Class module are shown in Figure 1.

A major benefit of the Pleiades architecture for the notional mission is that it decouples the deployment of the EO sensors from each other, enabling the customer to launch the most mature sensor and begin executing the mission immediately instead of waiting for the completion of the least mature sensor. To achieve this decoupling, we would house each of the three EO imagers in a separate Atlas-class module. The first launch

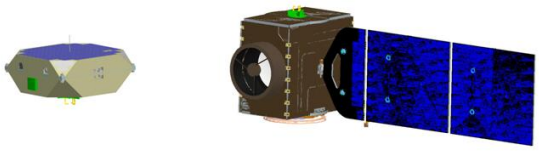

Figure 1. Pleione-class module (left) and Atlas-class module (right).

would carry the most mature sensor to allow time to develop the less mature sensor, and would also carry a co-manifested Atlas-class module with a data recorder and a high-speed downlink. The sensor-hosting module would also have a TDRSS transceiver for 24/7 TT&C coverage. This first launch is thus capable of executing part of the mission: it can collect, store, and then downlink large volumes of images to a dedicated high-speed ground station. After the first launch, suppose that the customer realizes that they also have in situ buoy-borne sensors that produce temperature data that they want to collect on a daily basis, and they want the imagers to provide images of ocean regions that show in situ measurements within certain temperature regimes. For the second launch, we would launch another Atlas-class module with the second, more mature, EO imager along with two Pleione-class modules - one with a computer for image processing and one carrying the both the receiver for buoy-based temperature data and a second TDRSS transceiver. The second imager module would be capable of simultaneously targeting the same ground point as the first imager when the cluster flies over a target, but they could also be independently targeted. The next launch would carry the third EO sensor, but would also insert an additional Atlas-class module equipped with a second data recorder and high-speed downlink. This will provide additional resources to the cluster and allow for future growth with the launch of additional payloads.

III.A. Notional fractionation

In the Pleiades fractionated architecture, each module must provide a minimum set of functions to be launched independently, join the cluster and maintain certain safety requirements such as collision avoidance. This minimum set of capabilities include those which are found in a typical spacecraft bus: attitude control, power management, thermal management, safe mode avionics, propulsion, and a TT&C link. Rather than attempting to fractionate these core subsystems, which would require significant technology breakthroughs and produce a questionable return on investment, we plan to develop a system that fractionates resources and payloads. A unique resource or payload would reside on each module and share the resources with the rest of the cluster. The technologies needed for this space-based resource sharing have been rapidly advancing as they are also critical in everyday use of the Internet. We see distributed computing as the major enabler of fractionated spacecraft, supported by wireless communications and packet networking. These technologies will be explained in section IV but first we will describe the resources that will be shared in our system.

Our notional system for launch supports missions for two independent stakeholders, and baselines the following fractionated capabilities:

- Mission payload: The payload or instruments that perform the intended mission (which can be the same in each module or different between modules) can be carried on one or more modules.
- Continuous communication: One or more modules will carry subsystems that provide near-continuous 24/7 TT&C connectivity for the cluster such as a TDRSS transceiver.
- High bandwidth downlink: While each module will have an AFSCN-equivalent space to ground link, cannot support high-bandwidth communications to the ground. One or more modules might need to provide a high-speed communication link with the ground to enable the downlink of large data volumes in short periods of time.
- Large volume data storage: Depending upon mission needs and the ultimate implementation of the communications requirements, one or more modules could carry a solid state recorder to store large volumes of data generated by the payload modules between ground contacts.

- Mission data processing: One or more modules could carry a high speed computer capable of performing complex mission data processing activities for the other modules in the cluster.

As shown in Figure 2, the completed Pleiades system will carry three precision pointing payloads for one stakeholder and a single coarse pointing payload for a second stakeholder. These mission payloads will share two 24/7 TDRSS telemetry, tracking, and command (TT&C) links, two solid state recorders, two high bandwidth downlinks, and two high performance mission data processors. The combination of function redundancy between modules and software to use it yields high availability. Pleiades will also provide infrastructure for future payloads beyond the F6 program.

Launch	1		2			3	
Module Name	Alcyone	Electra	Maia	Celaeno	Asterope	Merope	Taygete
Module Wet Mass (Kg)	225	225	225	75	75	225	225
Precision Pointing Payload	●		●			●	
Coarse Pointing Payload				●			
TDRSS	●			●			
Solid State Recorder		●					●
High BW Downlink		●					●
Mission Data Processor					●	●	

Figure 2. Resources are fractionated amongst different modules and shared throughout the system

We next consider a series of notional launches, each carrying a subset of the modules, and describe the mission capabilities provided by each launch.

III.B. Launch I: Alcyone and Electra: "Alpha Cluster"

The first launch will carry two Atlas-class modules named Alcyone and Electra (Figure 3) that will populate the "Alpha Cluster". The two modules will separate and acquire a safe relative parking orbit (SPO) with dimensions up to 2 km x 4 km. The two orbits will have the same semi-major axis, but will differ in longitude of the ascending node and eccentricity to prevent drifting out of range and the possibility of collision due to along-track intersections. Following successful on-orbit checkout, each module will use relative sensor-derived state knowledge (differential GPS and cross link ranging) to exit the SPO and enter V-bar station-keeping with a separation of 2 km. The cluster will maintain passive collision avoidance and active collision avoidance. Figure 3 also describes our approach to additional operations that are required as part of the DARPA System F6 demonstration such as wireless power transfer and back-door thruster commanding.

Alcyone will feature a precision-pointing payload and a TDRSS transceiver. Electra will contain a solid-state recorder and a high-bandwidth downlink. This first launch will demonstrate fractionation because payload data from Alcyone can only be downlinked at a usable rate through Electra. Additionally, the single TDRSS transceiver on Alcyone achieves continuous tracking telemetry, and command (TT&C) for the pair of modules.

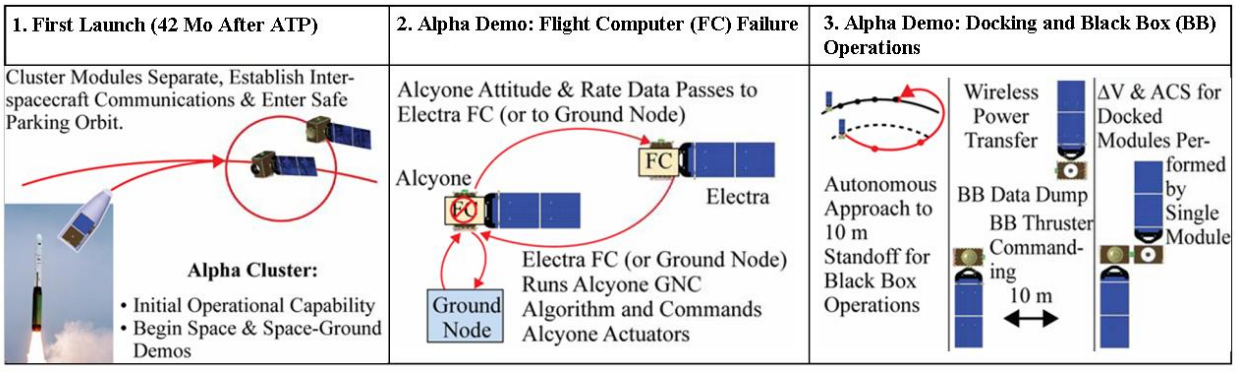

Figure 3. Alpha demonstration.

III.C. Launch II: Maia, Celaeno, and Asterope: "Bravo Cluster"

The second launch will carry one Atlas-class module named Maia and two Pleione-class modules named Celaeno and Asterope (Figure 4). Maia will carry a second precision-pointing payload for the first stakeholder. Celaeno will carry a second TDRSS transceiver and a coarse-pointing payload for the second stakeholder. Asterope will feature a high-performance computing mission data processor. Celaeno's TDRSS transceiver will serve as a backup for the initial cluster and might be used as infrastructure for future cluster expansion (beyond the initial seven modules). Asterope's mission-data processor will be much more capable than the flight computer common to all modules and will perform computationally intensive mission-related tasks such as real-time image processing. The Minotaur launch vehicle will place Maia, Celaeno, and Asterope in a close orbit to Alcyone and Electra. The new modules will separate and form a string-of-pearls sub-cluster in the same way that the Alpha cluster formed. Operators on the ground will command the sub-cluster to transition to an orbit that will drift within 10 km beneath the original cluster. Once the new modules cross the 10 km threshold, they will establish communications with the original cluster and their relative sensors will acquire lock. The new modules will autonomously gather with the original cluster and form the "Bravo Cluster." Additional cluster flight demonstrations such as geometry change and cluster safety are also described in Figure 4.

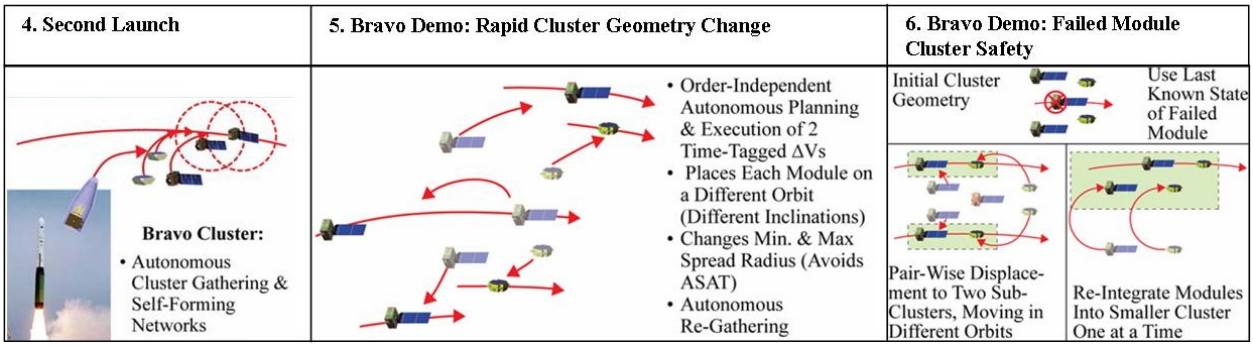

Figure 4. Bravo demonstration.

III.D. Launch III: Merope and Taygete: "Charlie Cluster"

The third launch will carry two Atlas-class modules named Merope and Taygete (Figure 5). These modules will join the cluster in the same manner described above and will complete the "Charlie Cluster." Merope will fly the third precision-pointing payload for the first stakeholder. Taygete will carry a solid-state recorder and a high-bandwidth downlink. The launch of Merope and Taygete will demonstrate system scalability as processing, storage, and downlink capacity for the system is increased. After the demo is complete, Pleiades will provide an infrastructure to continue its primary mission and supply resources for additional mission payloads.

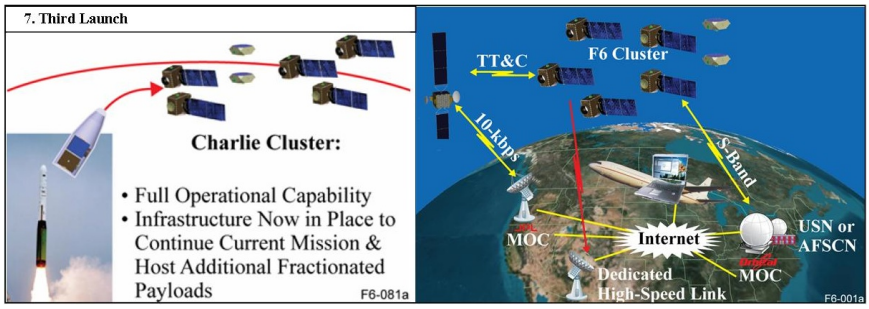

Figure 5. Charlie demonstration.

IV. Information integration with the Virtual Mission Bus

The Pleiades system achieves information integration with service-oriented software applications operating on Virtual Mission Bus (VMB) middleware over networks. The VMB supports failure detection and recovery within applications, as well as migration of pieces of an application from one processor to another (even between spacecraft modules or to the ground system). The middleware also provides resource management to ensure that each application has the resources it needs, and that two applications that share a processor do not interfere. In this section we will describe the goals and preliminary design of the VMB.

The principal architectural goal of the VMB middleware is to enable end customers to design and develop applications consistent with value-oriented engineering principles. The VMB delivers on this goal by providing the following key capabilities for applications.

- Flexibility: Applications can span a wide spectrum of end uses. For example, we are using the VMB as the foundation for the cluster flight application and the mission application.
- Adaptability: Applications can execute on a variety of underlying hardware configurations, i.e., different cluster and module configurations. For example, the mission application, by leveraging the VMB, will be able to make use of new sensor and CPU nodes as we insert those nodes into the cluster.
- Scalability: Applications can scale up to execute over one or one hundred nodes. We promote scalability in two main ways. First, we ensure that the VMB uses lightweight mechanisms to back the programming patterns we export to applications. Second, we we employ autonomous resource management and isolation algorithms to reduce the need for people to be a part of the normal operational loop, and to decouple the independent applications from each other.
- Reliability: Applications can continue to execute even after catastrophic unplanned changes in the underlying hardware configuration. Note that reliability is related in part to adaptability and to the isolation aspect of scalability.
- Security: Applications can meet the information assurance requirements defined by the end customer.
- Interoperability: Applications can run on modules that have different internal hardware revisions (such as new revisions of the same processors) or that are produced by different manufacturers.

The primary architectural abstractions of the VMB are to treat an application as a collection of cooperating *agents*, and the network-connected components within modules as resource-providing *nodes*. Figure 6 shows an abstract notional Pleiades cluster of modules. Each of the components—ACS processor, off-module wireless communication transceiver, general computing processor, and so on—is a node that can provide resources such as CPU cycles, memory, network bandwidth, and access to special-purpose sensor and actuator hardware. Each application—cluster flight, mission, and so on—consists of agents hosted by the nodes that provide the resources needed by the application. For example, mission application 1 has agents hosted by camera sensor nodes on Alcyone, Maia, and Merope, by computing processor nodes on Asterope and Merope, and by storage nodes on Electra and Taygete.

The notional cluster environment example demonstrates many of the key capabilities provided to applications by the VMB. The cluster as shown hosts two mission applications and a cluster flight application running concurrently. These applications can have separate end customers and controllers, with potentially different information assurance (IA) requirements. Moreover, these applications—particularly the mission applications—can safely share nodes, despite having different requirements.

IV.A. Node and network architecture

A node in the VMB provides resources to agents of applications. In Pleiades, each of the components in a module is a node, including both general-purpose computing processors and traditionally special-purpose components such as the coarse sun sensor, reaction wheels, IMU, and so on. In addition, ground elements, including the ground station, are also nodes. The node exports information on its available resources to enable applications to discover those resources, and arbitrates accesses to its resources by hosted agents to ensure that each agent receives the resources it needs and does not interfere with the operation of other agents. Figure 7 shows a notional subset of nodes in a Pleiades cluster. The VMB provides homogeneous

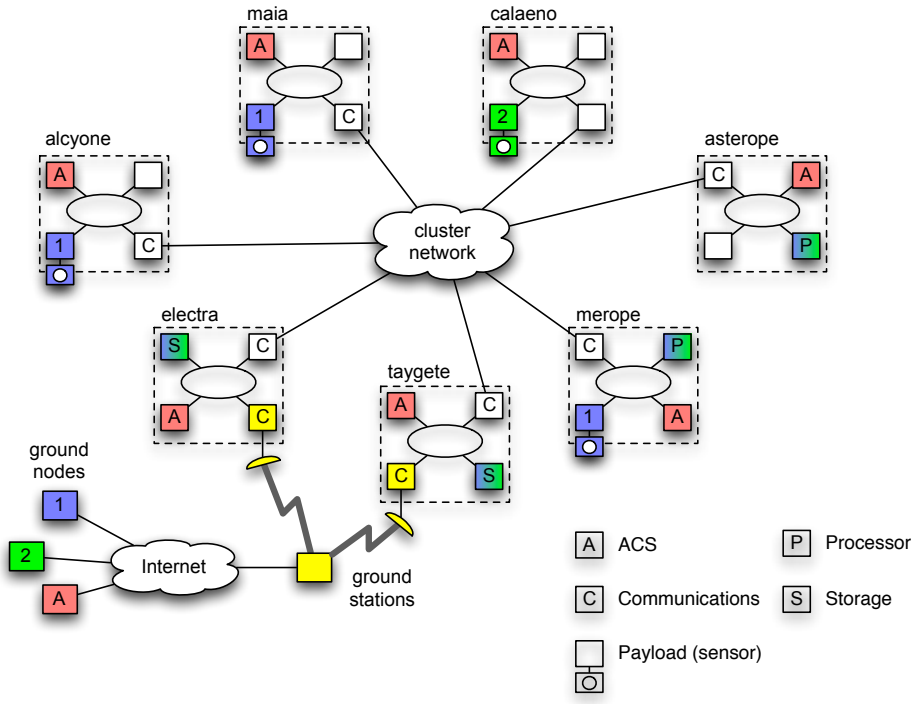

Figure 6. Notional cluster of modules in a Pleiades fractionated space system

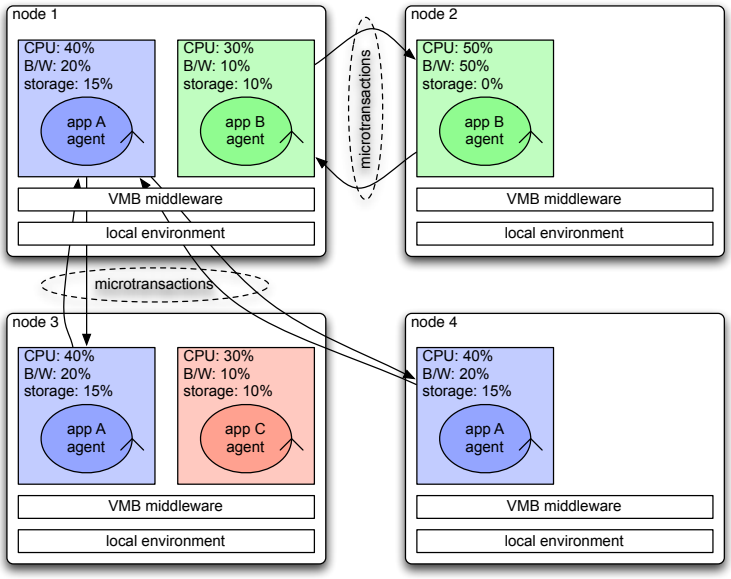

Figure 7. Notional subset of nodes sharing resources and communicating over the VMB

addressablility to nodes, and thus the nodes as shown may either be within the same module or be distributed across different modules.

A node manages the resources used by an agent through an abstract resource isolation container. For each agent, the node creates a container that enumerates the type and quantity of each resource used by the agent. For each type of resource, the node uses a resource-appropriate scheduler to arbitrate access by multiple agents to shared resources; for example, the node would use a real-time CPU scheduler to manage CPU usage, and a packet scheduler to manage network bandwidth usage. For even more stringent isolation on nodes that could host agents with different IA requirements, one could virtualize the resources and isolate each agent within a separate virtual node (analogous to a virtual machine), though we do not expect to implement such an approach in the current F6 program phase.

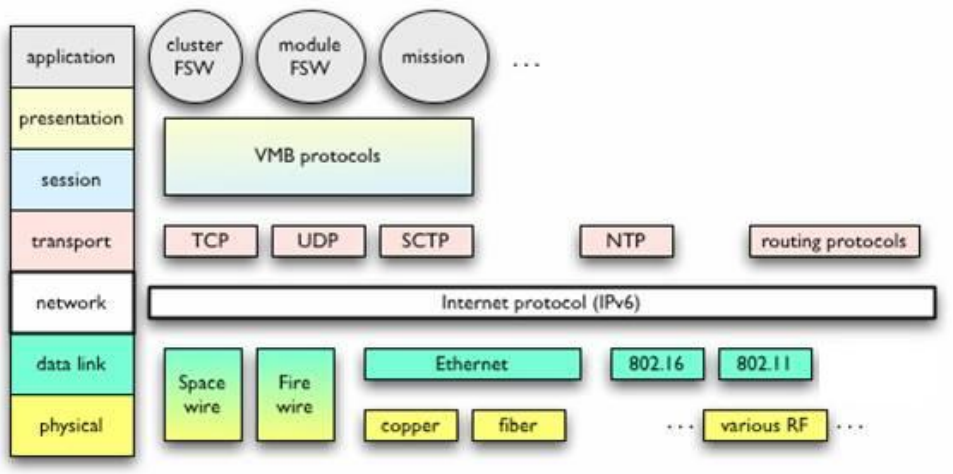

Figure 8. Notional network protocol stack at a VMB node.

The VMB leverages Internet Protocol (IP) network addressing and routing mechanisms to abstract away multiple discrete physical networks. A Pleiades system has three types of network: the intra-module network, the inter-module (cluster) network, and the space-ground links. IP addressing hides the relationship between a node and its host network, while IP routers move packets from one network to the other. A standard IP-based network protocol stack replaces the traditional custom interfaces and applications found in legacy spacecraft, allowing for the use and reuse of heterogeneous protocols at other layers and promoting network scalability. Figure 8 shows some of the applications and protocols available for use above or below the common IP layer in the protocol stack. The networking technology study has the goal of finding the optimal design and implementation of the IP-based network.

The node and network structure of the VMB enables the Pleiades architecture to fulfill a pair of key requirements for a fractionated system. The ability of each component on each module to communicate with any other component on any other module has the effect of unifying the individual free-flying modules into a single virtual monolithic spacecraft. Additionally, the ability of elements on the ground to communicate with components in space in similarly transparent fashion enables a robust, flexible and transportable ground station.

IV.B. Resource allocation and reservation

Resource allocation and reservation are decentralized processes in a VMB system. Any agent in the system with appropriate IA credentials can query nodes in the system for their resource types and availability, and reserve resources at the nodes to fulfill its requirements. Each node is responsible for advertising its available resources, and for receiving, admitting, and enforcing reservations for those resources. In this way, applications are no longer tied statically to specific components in a cluster, but instead may move dynamically to wherever appropriate resources are available.

An application in a VMB system uses one of a family of on-line resource allocation algorithms to map the resources that each of application agents requires to the resources that each of the nodes have. In general, an application 'creator' agent that starts the rest of the application agents performs allocation and reservation. The application creator requests resource types and availabilities from each node in the system,

and represents the reply as a vector that contains an element denoting availability per resource type. Each allocation algorithm takes as input a set of vectors that represent the requirements for each application agent, and a set of vectors that represent the availabilities at each node. In general, the algorithm then solves a linear programming problem that assigns each requirement vector to an unallocated availability vector, while applying some heuristic, e.g., best-fit (requirement uses most of the availability), or worst-fit (requirement uses very little of the availability).

IV.C. Microtransaction protocol

Agents of an application communicate with each other using a decentralized *microtransaction* protocol that guarantees atomicity and ordering for a set of operations from a sender agent to a group of receiver agents. Atomicity implies that either all operations within a microtransaction complete at all receivers, or none of them do. Ordering implies that all of the operations within one microtransaction from a sender have the same relative ordering with operations within another microtransaction (potentially from a different sender) at all of the receivers accessed by both microtransactions. Microtransactions provide a clean and lightweight way for an agent on one node to interact with agents on other nodes, without requiring close operational coupling (in particular, close clock synchronization) between the nodes.

Each sender tags its operations with a timestamp, and each receiver tags its data with the timestamps of the most recent "read" operation and the most recent "write" operation. The sender can generate timestamps from any monotonically increasing clock that is loosely synchronized to clocks at other agents; it does not have to rely on a globally synchronized clock, though a sender whose clock lags behind the other senders may find that its operations never succeed. The protocol is ultimately based on the the Palladio operation protocol[3] for coordinating reads and writes in a distributed storage application.

A sender can modify data on a receiver using a "write" operation. The write operation has two phases: a prepare phase and a commit phase. To perform a write operation on data distributed over a group of receivers, a sender selects a unique operation timestamp, and sends to each receiver a prepare request that includes the timestamp and the updates for the data on the receiver (note the desired updates may be different across receivers). A prepare request for some data succeeds if the timestamp on the request is greater than the timestamps of the most recent read and most recent write on the data, and if some other sender does not already have an outstanding prepare request on the data. If the prepare requests succeed at all receivers, then the sender sends to each receiver a commit request containing the timestamp. A commit request for some data succeeds if a corresponding prepare request exists; at that point, the receiver updates the timestamp of the most recent write on the data.

A sender can access, but not modify, data on a receiver using a "read" operation. The read operation has one phase: a request phase. To perform a read operation on data distributed over a group of receivers, a sender selects a unique operation timestamp, and sends to each receiver a request that includes the timestamp. A request for some data succeeds if the timestamp on the request is greater than the timestamp of the most recent committed write on the data, and if some other sender does not already have an outstanding prepare request on the data; at that point, the receiver updates the timestamp of the most recent read on the data.

IV.D. Group membership protocol

The group membership protocol provides mechanisms for aggregating the agents of an application into a manager-worker structure and for detecting the failure of agents. In general, the worker agents of an application perform distributed, concurrent operations, such as parallel computations. The manager agent of an application provides a virtual centralization point to perform application-wide operations, such as managing application-specific metadata, restarting failed workers, or reallocating workers to different nodes after a node failure. A manager is distinguished from workers by having only soft state; in the event that a manager fails, any new manager can construct its state from the workers' state.

A manager must manage a 'quorum' of workers at all times. The definition of a quorum is specific to the application and the assumed types of agent failures, but in general a quorum is at least more than half of the expected number of workers. If a manager controls less than a quorum, it must stop running. Otherwise, in the event that a network fault partitions the set of workers into two sets that cannot contact each other, the old manager risks managing a minority partition of workers while a new manager takes over managing the majority partition, which may lead to the two partitions taking inconsistent execution paths.

The group membership protocol includes a heartbeat mechanism that enables managers and workers to detect failures. Each worker asks its manager for a lease with an expiry period. If a worker fails to request a new lease at the end of the old lease period, the manager assumes that the worker has failed. Conversely, if a manager fails to respond to a request for a new lease, the worker assumes that the manager has failed. Having unexpired leases outstanding at a quorum of workers is equivalent to managing a quorum of workers.

The group membership protocol also includes an election mechanism in which 'candidate' agents compete to become manager. Elections occur when the set of workers first starts up, or when the manager fails. Each candidate tries to gather as many workers under its control as possible. If a candidate has already acquired a worker, the worker will reject acquisition requests from other candidates by responding with its current candidate, which incidentally enables candidates to learn about each other. A candidate that gathers a quorum of workers wins the election, and transfers the workers of losing candidates to its control. The candidates maintain a ranking among themselves to break ties; in the event of a tie, a lower-ranking candidate will release its workers to the highest-ranking candidate.

V. Conclusions and future work

The Pleiades fractionated space system architecture has the potential to transform national security space missions through information integration. We expect that distributed, continually evolving fractionated space systems will replace the legacy paradigm of large, long-term development, all-or-nothing monolithic spacecraft. We have presented the basic framework of the system we plan to develop and demonstrate. We intend for Pleiades to be the beginning of a new system of systems in space, rather than just a technology demonstration. We plan to regularly publish papers and develop interface standards so that government and industry can easily and economically adopt this approach.

References

[1]Brown, O. and Eremenko, P., *The value proposition for fractionated space architectures*, AIAA SPACE 2006 Conference, Sept. 2006.

[2]Brown, O., Long, A., Shah, N., and Eremenko, P., *System lifecycle cost under uncertainty as a design metric encompassing the value of architectural flexibility*, AIAA SPACE 2007 Conference, Sept. 2007.

[3]Golding, R., *The Palladio access protocol*, Technical report HPL-SSP-99-2, Hewlett-Packard Laboratories, Palo Alto, CA, 1999.

[4]Golding, R., and Borowsky, E., *Fault-tolerant replication management in large-scale distributed storage systems*. 18th Symposium on Reliable Distributed Systems. Oct. 1999.