

# **Leveraging the Space Plug-and-Play Avionics (SPA) Standard to Enable Constellation-Level Collaborative Autonomy**

**Louis Marketos**

*Design\_Net Engineering*

*16080 Table Mountain Parkway, Suite 500, Golden, CO 80403*

*(303) 462-0096*

*lmarketos@design-group.com*

## **ABSTRACT**

On-orbit autonomy depends on the timely availability of situational awareness data. Data provided from ground stations, derived from telemetry from other spacecraft, need to be evaluated in conjunction with the real-time telemetry available from various subsystems on the spacecraft bus. Having direct access to telemetry from other spacecraft dramatically increases an autonomy engine's ability to make decisions based on near-real-time information and information from multiple sources with heterogeneous capabilities.

SPA is well suited to provide a platform to support spacecraft autonomy on several levels. It provides an abstraction layer between the autonomy engine and each spacecraft subsystem. This decoupling allows simplified reuse of the autonomy engine on future missions as well as reducing design-time efforts to develop the bus since interfaces between the bus and the autonomy system are standardized.

Introducing a Constellation Collaboration Manager (CCM) activity agent as a SPA-compliant application on the bus allows telemetry from other spacecraft, including telemetry from component types not available on the local spacecraft, to be available as inputs to an autonomy engine in the same abstracted manner that local telemetry is available. This allows an autonomy engine to base its decisions on a much broader range of data and increase the range of scenarios that it can handle. The telemetry available can be optionally pre-filtered or processed into aggregate information either by the CCM on the spacecraft broadcasting its telemetry or on the receiving spacecraft before it is sent to the autonomy engine. The CCM also opens additional options for autonomy engines by allowing them to generate tasking requests for other spacecraft in addition to its local spacecraft. Since the CCM is a SPA-based component, it can be added to an existing SPA-based system with minimal or no impact to other components on the bus.

## **INTRODUCTION**

On-board satellite autonomy leads to many benefits. From defensive counterspace applications to simply easing the burden of ground station personnel, an increase in autonomy results in an increase in responsiveness, survivability and utility. The quality of output produced from any autonomous system is directly dependent on the timeliness, quality and quantity of data available as inputs. Spacecraft built to support the SPA standard can take advantage of its data-driven architecture to allow autonomy engines access to the telemetry they require through a standard interface that is decoupled from the specific implementations of application and device drivers in the rest of the system. This decoupling is the key piece of the SPA standard that supports software reusability among different

spacecraft and missions. It also turns out to provide a framework that is relatively easily extended to support data sharing, not just among components on a single spacecraft, but among applications executing on different spacecraft.

Sharing data among spacecraft in a constellation dramatically expands the domain of data that an on-board autonomy engine can consider when making decisions. This results in not only increasing its ability to make situational awareness decisions but also to take advantage of decisions already made by other spacecraft and react accordingly.

### SPA OVERVIEW

To understand the simplicity and elegance of the role SPA can play in support of constellation-level autonomy it is important to have at least a cursory understanding of SPA as it relates to spacecraft flight software. This is the focus of this section. Note that this section is not a complete overview of the SPA standard.

The primary goal behind the development of the SPA standard is to support rapid spacecraft development. For flight software, this is accomplished by defining a framework that allows each software component to be cleanly decoupled from other applications and components on a spacecraft. A publish-subscribe design pattern is used with the addition of a broker application called the Data Manager (DM). The DM is one application in a software suite called the Satellite Data Model (SDM) that provides a reference implementation of the SPA standard.

Each software application and hardware device on the spacecraft has an eXtensible Transduced Electronic Data Sheet (xTEDS). xTEDS are essentially Interface Control Documents (ICD) that captures the identity, role, parameters, and services that a component provides. The DM collects xTEDS from components, hardware and software, and uses them to keep an inventory of which components offer which services and data. The DM can then match consumers of data with the appropriate producer. As illustrated in Fig. 1, Applications query the DM to locate the information required and if a source is found, a subscription is created allowing the component to publish its data directly to any other component in the system that is interested in it.

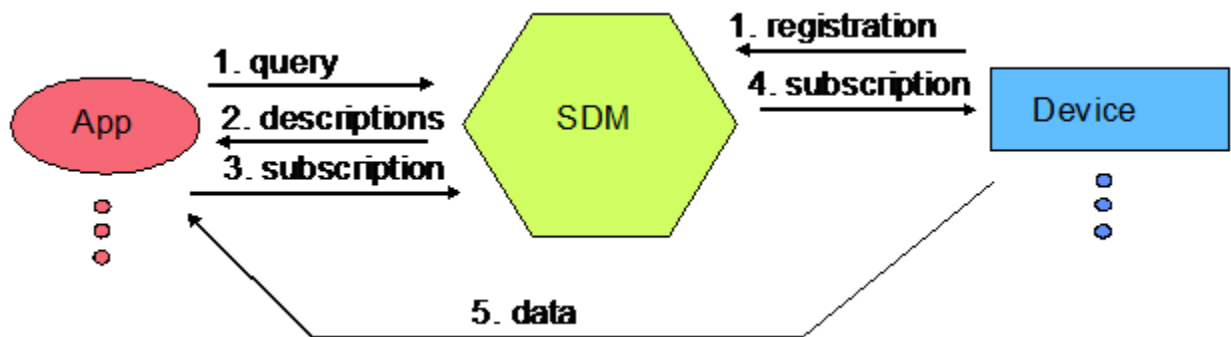


Fig. 1. SPA Data Decoupling

The use of xTEDS and the role of the DM as the information broker perform an extremely important function in the overall SPA vision. They allow hardware and software to interact in a decoupled manner. This results in hardware devices and software applications being treated identically from a data perspective. If an application subscribes to data, there is no difference if it is getting that data from a device or an application that may be publishing processed data. This abstraction is critical to several aspects of SPA including enabling the reusability of individual software applications and supporting Flight Software In the Loop (FSWIL) testing, a method of testing the system using a

software simulator connected with instances of actual flight software running on either flight processors or other computers illustrated in Fig. 2.

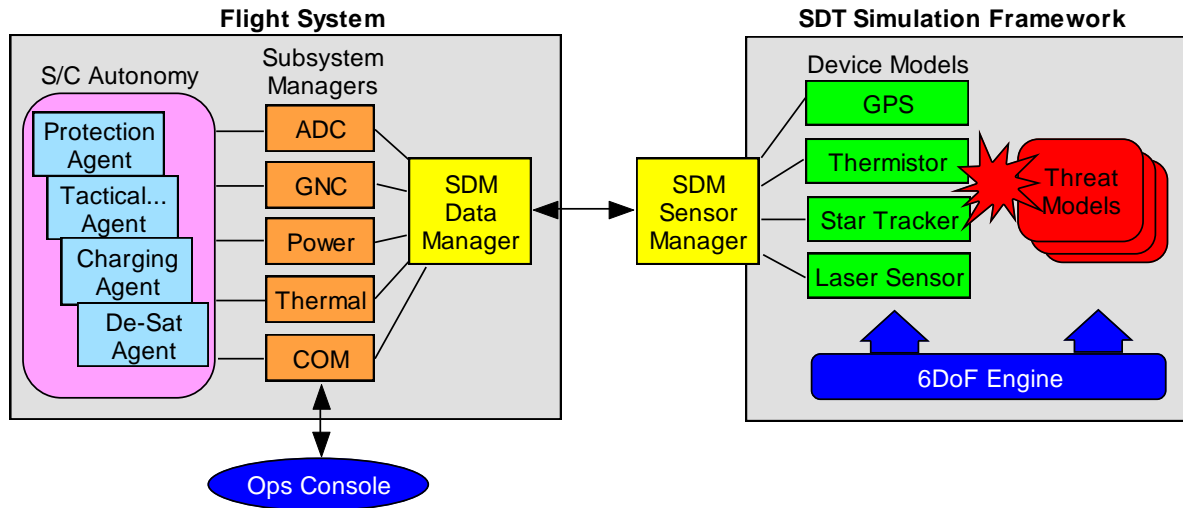


Fig. 2. SPA Support for Flight Software In the Loop (FSWIL) Testing

In the context of this paper, the decoupling of spacecraft components has two additional benefits. First, it allows interfaces to autonomy engines to be abstracted in exactly the same way other components in the system are. Second, interfaces between spacecraft in a constellation can be treated in the same way as interface among components within a single spacecraft.

The use of xTEDS to define hardware and software interfaces has other benefits as well, not the least of which is natural integration into design tools such as Design\_Net's Mission Spacecraft Design Tool (MSDT). As shown in Fig. 3, using the information included in the xTEDS as well as other metadata, MSDT can take high-level mission requirements and generate spacecraft configurations to meet those requirements. Additionally, MSDT can generate all of the required files to run a simulated mission in a tool such as Star Technology's Satellite Design Tool (SDT).

To run a spacecraft configuration through simulation, MSDT distributes the configuration of the spacecraft to the simulation framework, along with operational scenario details such as orbit, location of threat sources, faults to simulate, and others. At the same time, the flight software for the satellite, consisting of the basic modular support code as well as platform-level autonomy applications can be distributed to the flight system or flight system proxies representing each satellite. Sensor Manager software hosted by the simulation framework then allows the flight systems to interact with virtual spacecraft devices in coordinated simulation. The Sensor Manager interacts with the flight software using the SPA protocols so there is no impact or special support required from the flight software. This infrastructure, illustrated in Fig. 2, allows coordinated autonomous software to be developed and evaluated in a rapid and repeatable test environment. This ability to rapidly change spacecraft configurations and evaluate the impacts of those changes quickly is a key advantage of using SPA to implement the features described in the remaining sections of this paper.

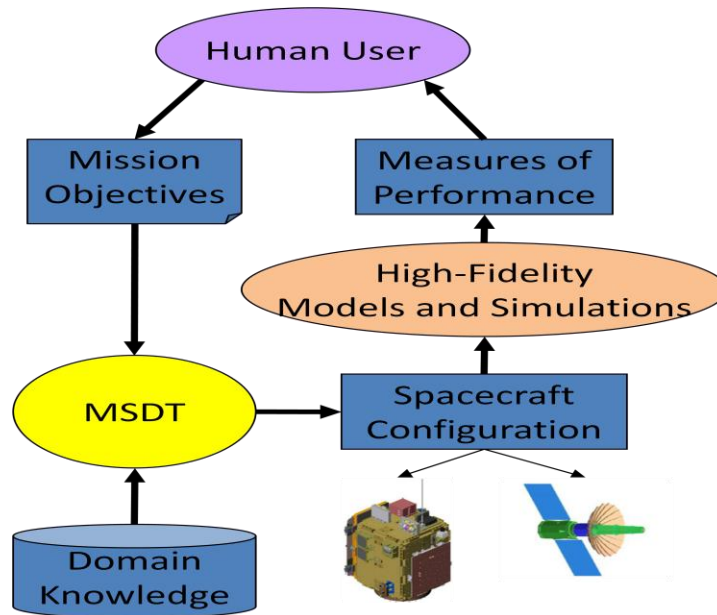


Fig. 3. MSDT Process Flow for Spacecraft Design and Simulation

### SPACECRAFT AUTONOMY USING SPA

Autonomy has many benefits to tactical space operations, not the least of which is the ability to transition a portion of the typical command and control aspects of satellite operation from central ground operations to the platform itself. As flight processors become more capable and it is possible to have greater degrees of high-power computing on-orbit, a natural course is to perform on-orbit processing of the data collected from sensors. One logical use of that data is to feed it to on-board algorithms that fuse the data and perform processing to detect and identify threats to the spacecraft. Response models that are resident on-orbit have promise for streamlining the process that protects our tactical space assets.

The ability to perform this processing on-orbit has several advantages. Timeliness – identification and response is rapid for threats that can be detected using data collected solely by the individual satellite because the latencies of the space-to-ground communications link are not present. Decisions can always be made as quickly as the available processors can perform their work. Efficient allocation of fusion-oriented processing load offloads that responsibility from the ground sites. Fleet decision support can be made much more responsive and efficient with this model. When a single space platform produces a large amount of sensor data, all of which could be potentially fed to complex fusion algorithms, it is easy to imagine that the space-to-ground link rapidly becomes a bottleneck in the system, especially when other operational exchange of telemetry and commanding is also required.

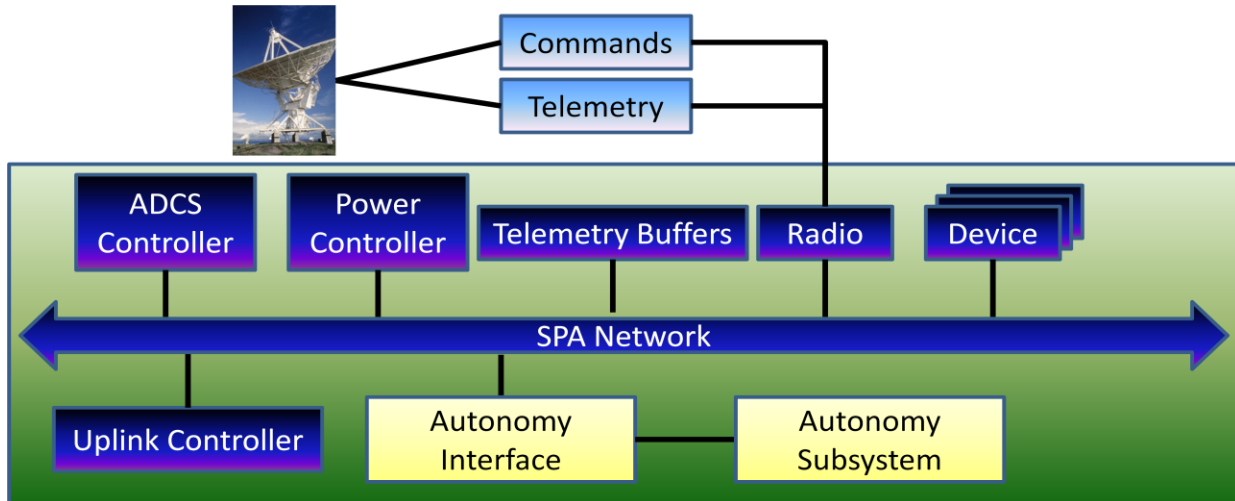


Fig. 4. Autonomy Engines in a SPA-based System

The key to developing a more efficient and capable decision support system is to address core capabilities at the space platform level. SPA’s data-centric approach to component communication is ideally suited to support autonomy engines as plug-and-play components. As shown in Figs. 4 and 5, by isolating the autonomy engine from the specifics of the spacecraft bus it can focus on collecting the data required for it to make appropriate decisions and results in a more reusable component. On the other hand, from the bus perspective, decoupling the autonomy engine allows multiple engines to be efficiently evaluated at design time on identical busses and using SPA’s FSWIL capability they can be evaluated across identical scenarios and simulated sensor input.

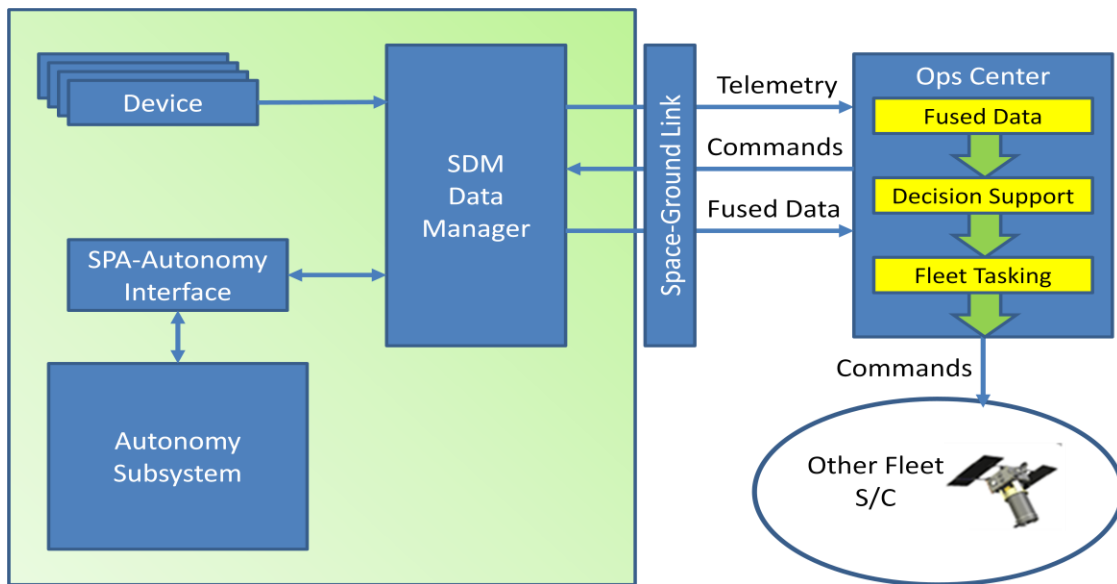


Fig. 5. SPA Support for Fused Data to Drive Commanding of Other Spacecraft

The use of design-time tools such as MSDT can simplify the process of configuring and evaluating different autonomy engines even further. MSDT supports custom scripts that can be used to preconfigure autonomy engines with the learning data and initialization state information they require to be tailored to various scenarios. This is a natural step that occurs as part of the spacecraft configuration generation and feeds the FSWIL simulation with appropriate data to evaluate the response of the autonomy engine.

## CONSTELLATION COLLABORATIVE AUTONOMY

Having demonstrated the natural fit between SPA and autonomy on a single spacecraft, an obvious next step is to apply the same principles to multiple spacecraft simultaneously. In the same way that a simple bridge between autonomy subsystems and the components on a SPA bus allow robust interaction among components on a spacecraft, a slightly more complicated but still relatively straight forward bridge, called the Constellation Collaboration Manager (CCM), enables interaction among SPA-based spacecraft. Fig. 6 shows the CCM in the context of a SPA-based constellation. The level of detail available for interactions can be scaled based on quality of service requirements and space-to-space network characteristics. Embedding xTEDS-like metadata as part of CCSDS formatted messages among spacecraft allows discovery of capabilities, notifications of events and availability of status information down to a fine grade of detail if required.

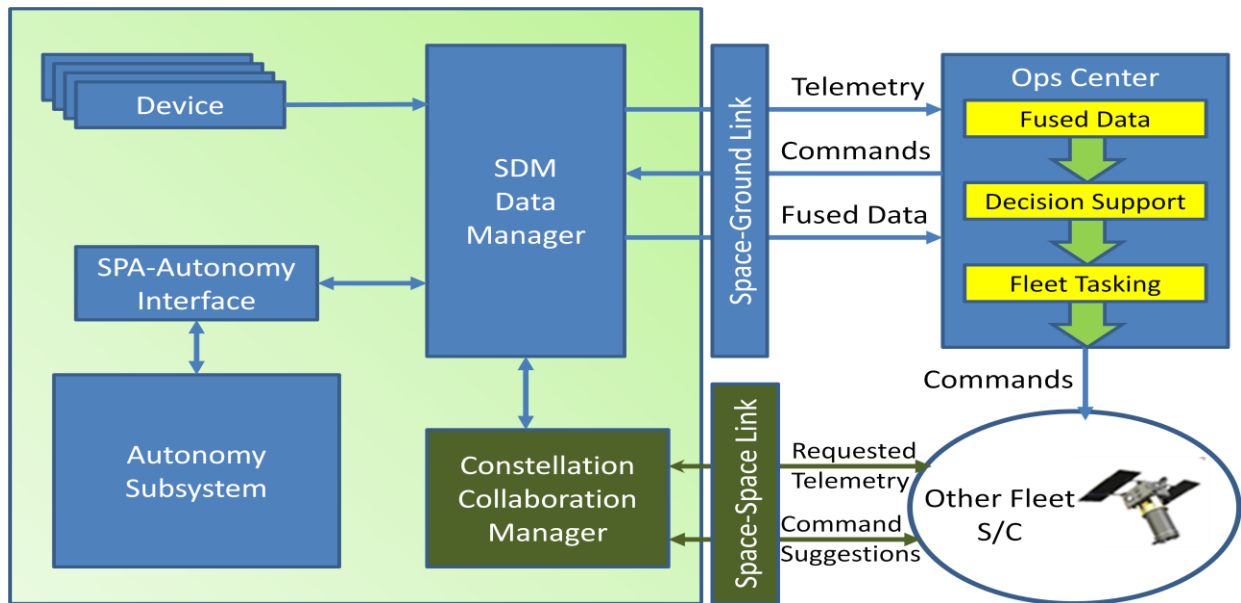


Fig. 6. The Constellation Collaboration Manager Exposes an xTEDS-like Interface among Spacecraft

Allowing autonomy engines to subscribe to data not only from the spacecraft they reside on, but also from other spacecraft in the constellation opens up new opportunities for data fusion that would otherwise only be possible using ground-based assets. Additionally, possibilities to utilize distributed computing techniques allow a spacecraft whose processors are overburdened to potentially offload some processing tasks to underutilized processors on other spacecraft as well as taking advantage of independently determined situational awareness information from other spacecraft...all without requiring interaction with a ground station.

This interactive capability among spacecraft is referred to as “Collaborative” to emphasize the distributed nature of the autonomous decisions being made. While the autonomy engine on any given spacecraft can publish its results or even suggest a course of action to another spacecraft, it is up to the engines on other spacecraft to take those results and recommendations into account and determine their own course of action. Another aspect of the collaborative nature of a constellation of such spacecraft is further emphasized in the section “Tactical Tasking” later in this paper.

Similar to how MSDT can support evaluation of autonomy engines and spacecraft configurations as described in the previous section, it can play an important role at design time in the evaluation of the effectiveness of a constellation of spacecraft at fulfilling a given mission or set of missions. It has the ability to take a high-level mission requirement and generate a constellation of spacecraft configurations. The satellites may be identical functionally or

may each possess unique functionality. The configuration of each satellite can be distributed to the simulation framework as well as loading the flight software into either a flight processor or other computer. The simulation then starts multiple Sensor Manager, shown in Fig. 7, one for each spacecraft that allow the flight software to interact with independent virtual spacecraft devices during simulation.

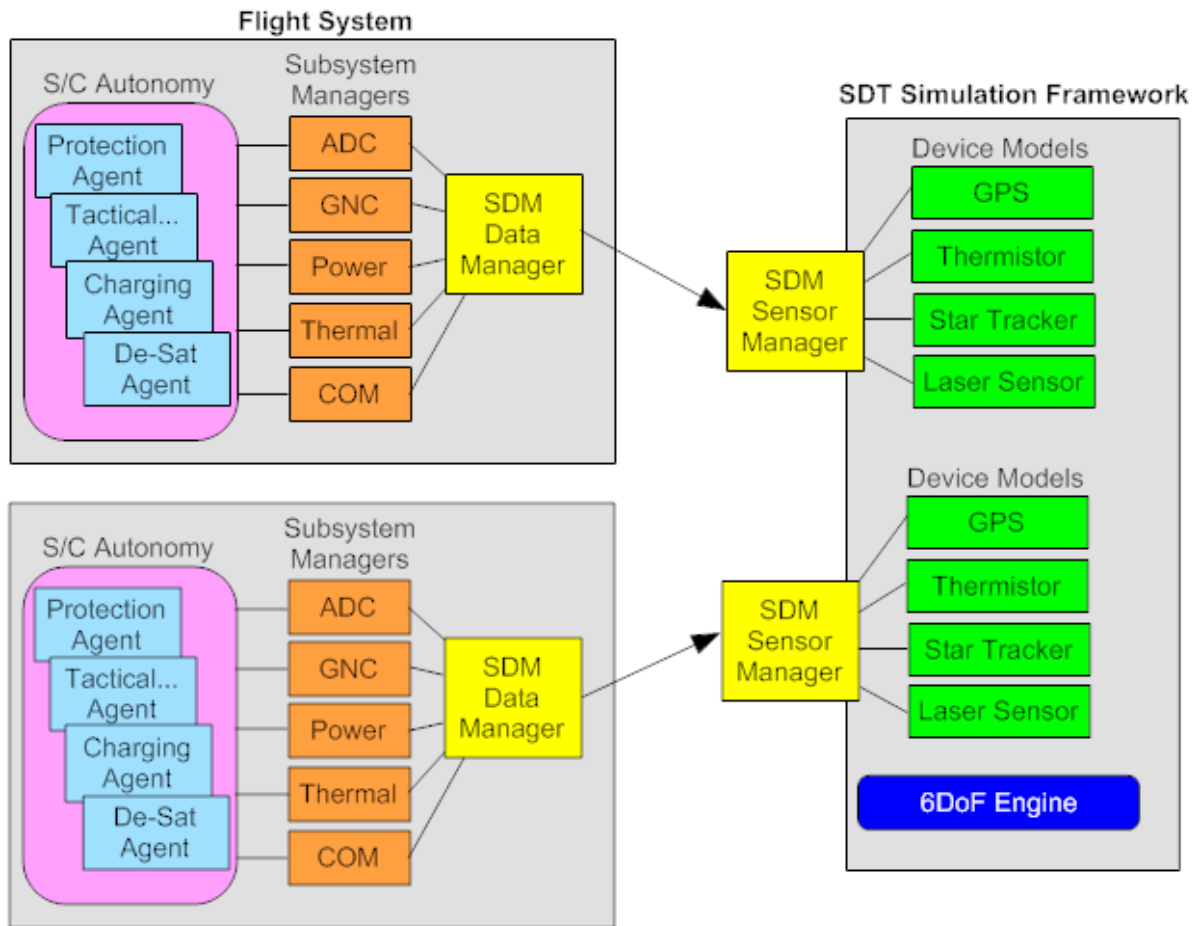


Fig. 7. Multiple Sensor Managers Enable FSWIL Testing of Constellations

## TACTICAL TASKING

A somewhat surprising but natural extension of constellation-level autonomy is support for constellation-level tasking. By treating a constellation of spacecraft as a single task-able entity, the ground-based effort of developing and planning tasking can be greatly simplified. This simplification combined with SPA's goal of supporting rapid, inexpensive development of tactical assets enable a tactical user to task the constellation in a very natural way. Using a straight-forward arbitration algorithm and on-board what-if evaluation capabilities allow a constellation to self-negotiate which spacecraft can best handle a given task given parameterized priorities such as timely completion, quality of result, and others. A simple example is illustrated in Fig. 8. The tactical user first issues a request to any spacecraft in the constellation he has access to. That spacecraft then queries other spacecraft to evaluate how well they predict they can fulfill the task. The spacecraft that can best satisfy the request is assigned the task and executes it at the appropriate time. Extensions for more complicated tasks, composite tasks, and

fallback capability to handle cases when spacecraft cannot complete tasks are handled in a similar manner by the CCM.

With detailed mission planning occurring on-orbit among the various spacecraft in a constellation, a tactical user can focus on higher level mission goals and not be required to know or understand the technical strengths and weaknesses of the payloads available on each spacecraft in the constellation.

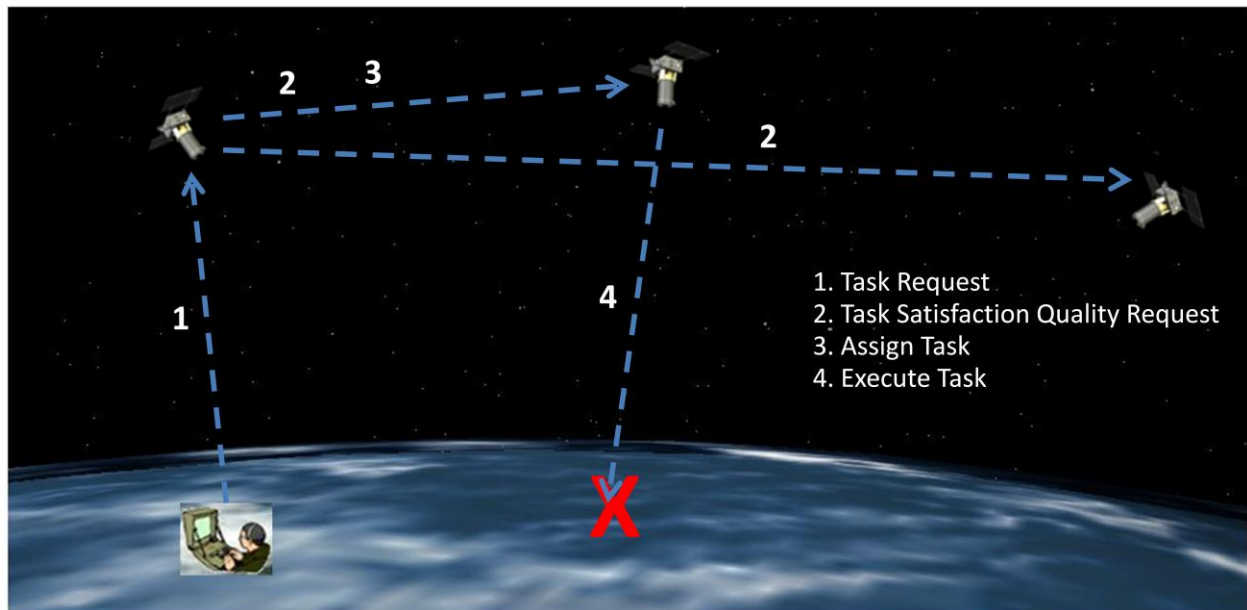


Fig. 8. Sample Tactical Tasking Execution

## CONCLUSION

As in terrestrial computing, the use of basic computer science principles in the design of software architectures such as decoupling and object brokering has a wide range of advantages. Spurred on by the availability of more capable flight processors, technologies such as SPA are able to leverage computer science techniques that have been proven over many years in terrestrial computing environments. These advances have made it possible to add new functionality to spacecraft, increasing the capabilities of individual platforms as well as allowing possibilities for distributed processing that maximize the available processing power. The capabilities described in this paper, decoupled autonomy engines, distributed collaborative autonomy and tactical tasking of constellations while very promising in terms of increasing on-orbit performance are just the first generation of techniques to take advantage of the potential of SPA.