

# **SNARE (Sensor Network Autonomous Resilient Extensible): Decentralized Sensor Tasking Improves SDA Tactical Relevance**

**Robert Carden**

*The MITRE Corporation, rcarden@mitre.org*

**Dustin Burchett**

*The MITRE Corporation, dburchett@mitre.org*

**Harvey Reed**

*The MITRE Corporation, hreed@mitre.org*

## **ABSTRACT**

Increased volume of space traffic and number of Resident Space Objects (RSOs) strains sensing resources and requires increased coordination and automation of Space Domain Awareness (SDA). The Space Surveillance Network (SSN) currently utilizes a network of over 30 ground-based radars and optical telescopes in addition to on-orbit telescopes to provide observations on a catalog of greater than 20,000 RSOs. These observations help detect, track, identify, and catalog RSOs which informs SDA. Due to orbital perturbations, RSO observations need to be periodically refreshed, else the last RSO Two Line Element Set (TLE) may not represent the current orbit.

Presently, the Space Defense Operations Center (SPADOC) employs the system known as the Special Perturbations (SP) Tasker to help maintain catalog accuracy. The SP Tasker provides a daily (but coarse) schedule to the SSN, which in turn provides information in the form of sensor observations to aid in TLE refreshes. SP Tasker is limited by several key factors: SP Tasker tasks once per day and hence cannot automatically adapt to events as they unfold, and, the premise of tasking via the SP Tasker is that events are “fixed,” which is not generally the case.

SNARE (Sensor Network Autonomous Resilient Extensible) is a United States Space Force (USSF) effort which decentralizes the tasking of, and collection from, SSN sensors which report to the Combined Space Operations Center (CSpOC). SNARE seeks to transform the current catalog-keeping functionality of the SP Tasker into a real time positional data stream, enabling each sensor to self-task based on reading trusted, decentralized, information. This informs SDA tactical relevance based on improved timeliness of positional information and positional accuracy.

SNARE is composed of sensing nodes (sensors), sensor interfacing nodes, and information validating nodes. Interfacing nodes measure available, up-to-date data to produce “utility” of collect values for each RSO. This is done primarily using the Local Value Function (LVF), which can rank order every object in the catalog to what is important to collect on “now.” The LVF takes in account requirements of each RSO coupled with the known, up-to-date situational awareness of each RSO. Important objects generally have higher collection values based on their requirements, and objects that go unseen for too long or have high uncertainty in their positional state have higher collect values at a given point in time. Since the LVF is a continuous, deterministic, function over time, each sensor independently arrives on what needs to be tasked on currently, and what can feasibly be done.

Sensors have the responsibility of taking the current ordered list of objects throughout the day to collect on and then determine (in a locally optimized sense) what should be currently collected on to maximize network utility. Sensors then provide the interfacing nodes with observations which are then validated and processed by the validating nodes. These nodes inevitably create or refresh TLEs and provide other situational awareness--which is up-to-date information that is then available to the network for future collects. Validating nodes also prevent redundant collections between sensors. The CSpOC reads the processed SNARE positional state, including collections as they occur, providing CSpOC a real time positional stream of data to further process and publish.

The SNARE positional data stream in near real time enables the resulting SDA to be tactically relevant, where SNARE monitors and responds to events as they unfold. This tactical relevance is a direct result of decentralizing and automating tasking functions which are presently centrally performed in CSpOC. Trusted data from SNARE enables an information provider and consumer relationship between SNARE and CSpOC.

The SNARE effort has completed initial modeling and prototypes with rationale for improving SDA tactical relevance and is presently transitioning to operational prototype phase.

Key findings are that (a) once decentralized, the independent decentralized sensors can make decisions on their own, given access to trusted shared data across the SNARE network, and (b) the decentralized sensors can coordinate using data sharing and provide an improved emergent information gain (sensors influence each other). This paper describes the results of modeling and prototypes, followed by the approach for data sharing in the SNARE network. For example, the prioritization and collection decision processes are described in detail, providing the reader with an understanding into how certain challenges (e.g., legacy Special Perturbations (SP) Tasker once-per-day tasking), can be addressed with decentralization yielding an improved real time positional stream of data. The paper concludes with considerations for information sharing beyond positional data, enabling higher-order functions, such as the BESTA (Blockchain Enabled Space Traffic Awareness) and GREAT (Global Resilient Extensible Autonomous Trusted systems) research efforts.

## 1. INTRODUCTION

The central problem with tracking space objects is that space is volumetrically huge (estimated to be 73 trillion cubic miles from Low Earth Orbit [LEO] to Geosynchronous Earth Orbit [GEO]), the space catalog size is large and rapidly growing, and sensors have limited resources to observe this large volume and quantity of objects. Scheduling a set of sensors to observe each object can be NP-hard, as is observed in the classic traveling salesman problem and in the operational SP Tasker algorithm used by the United States Space Force (USSF) for the SSN sensors.

Complicating the situation are numerous interests of many stakeholders who are responsible for both providing capabilities using satellites through a lifecycle (e.g., launch, orbit, maneuvers, end of lifecycle decommission and possibly deorbit), and users of these capabilities (e.g., governments of nations across the globe, commerce, academics, and civilian). The value of the capabilities is enormous and is now a key component of modern daily life. For example, communication, GPS, tracking, weather, agriculture, rush hour traffic, all rely on space-based capabilities that require the freedom to operate in space. However, freedom to operate requires up-to date positional state of objects in the space domain. Therefore, accurate knowledge of the current and predicted positional state of satellites is critical to nearly everything we do on the planet.

Without the real time positional state of objects in the domain, we do not know where things are, where they are going, and more importantly their conjunctive relationship to each other. Freedom to operate requires not only trusted historical positional data of space objects, but also requires this information be as close to real-time as possible. The reason is that orbits change due to maneuver, and an old observation increases the likelihood that it is no longer the correct orbit. Without freedom to operate, modern daily life itself is at risk.

Producing a globally optimized schedule of sensor observations (via SP Tasker) is NP-hard, and presently experiencing limitations due to increasing number of objects, and increasing numbers of maneuvers, each of which breaks the observation schedule (sensor tasker) for that object. Thus, SMC/SPG Global Sensor Watch (GSW) is exploring improvements and alternatives.

The SNARE team pursued a reframing of the problem to convert the problem:

- From: a brittle (sensitive to maneuvers) globally optimized schedule for all sensors for a complete day produced by SP Tasker. The result is a brittle and computationally NP-hard problem.
- To: a concurrent locally optimized prioritization scheme, which is only concerned with choosing the next observation for that sensor, at that time, while checking to avoid duplicating a recent observation from another sensor (via shared data store). The result is a simple computation problem for each sensor that yields an emergent effect across the network of SNARE enabled sensors.

This paper describes the motivation and history of MITRE's SNARE prototype, key results during development leading to the initiation of the operational prototype phase, and a summary of the concept.

## 2. SNARE CONCEPT HISTORY

The SNARE concept was inspired by the need for SMC/SPG GSW to pursue positional awareness of objects in space in real-time or near real-time. The SP Tasker is the current regimen in place for the SSN to accomplish this goal. SP Tasker is a scheduling algorithm that is computed once per day, which analyzes information collected by the network in previous days and outputs a sensor schedule for collecting further information the next day. The SP Tasker, however, is limited by both the cadence at which it is executed and the approach of creating a global schedule. Such global schedules are computationally expensive and are based on untenable assumptions such as fixed orbits (no maneuvers since the last schedule). SNARE is conceived to be a computationally simple and deterministic prioritization algorithm that allows SNARE-enabled sensors to act autonomously (coordinating with shared real time data) to make collections throughout the day. The result is an emergent network effect of reactive and dynamic sensor observations, versus a static and brittle global sensor schedule.

MITRE studied examples of autonomous systems that cooperatively coordinate to achieve emergent effects: examples include cell phone networks, Smart Grid, and cloud compute resource loading. These examples use emergent effects to solve what is otherwise computationally hard problems. The SNARE approach similarly uses a network emergent effect to provide maximum information gain based on sharing near real time positional state information across the SNARE network. SNARE provides autonomous sensor observation tasking (via prioritization), and post processing such as change detection. The emergent effect includes dynamic tip and cue resulting from maneuvers to be detected and post-processed and shared as they happen. Further, by using a network approach, SNARE is extensible, and sensor agnostic architecture. Thus, SNARE autonomously compensates for the addition or removal of sensors from the network, which can also inform a sensor status function.

The initial proof of concept began in FY17 to evaluate whether or not logic could be developed to achieve the GSW objective. The MITRE team developed a modeling and simulation scenario to compare the SP Tasker and SNARE approaches. Each paradigm (via simulation) was supplied with the same sensor network and capabilities, RSO catalog, and pre-defined maneuvers for objects within the catalog. The metrics of comparison were:

- Positional accuracy: defined by comparing the positional results of each paradigm versus “truth” position (in this case, due to the nature of the positional data itself, 18 hours from “epoch” time was considered).
- Reacquire time: the first time the object was observed after it maneuvered (this can also be equated to the first time that the paradigm would have noticed change in positional state).
- Recovery time: the time it took to get three tracks on the object after it was reacquired.
- Gap characterization: USSF metrics to compare tasking methodologies over time using requirements that dictate the allowable “gap” in between collections across each object in the catalog.
  - Total Violation Time (TVT): The accumulated time an object fails to meet a specified revisit requirement.
  - Revisit Rate Interval (RRI): The rate at which an object achieves ‘useful’ revisits based on the 95<sup>th</sup> percentile gap time.

SNARE has used the above metrics to establish the meaning of “SDA tactical relevance.”

The initial pathfinder showed significant improvements to the metrics. It is noteworthy that due to the simulation environment and scope, tip and cue (a critical component of SNARE) was **not** enabled for experiment. The next experiment was to determine how SNARE performed at volume for change detection and recovery. A similar modeling and simulation scenario was set up to quantify the control aspects of SNARE, and its ability to recover from large numbers of object maneuvering at the same time. There were several increases to maneuvering objects culminating in 1,000 objects maneuvering at once. SNARE considerably outperformed the SP Tasker in all test cases under the same conditions. In summary, the analysis of SNARE over SP Tasker with the 1,000-object test case showed that SDA tactical relevance has been improved via:

1. Improved orbital state accuracy: Average accuracy of catalog objects improved by 0.8 kilometers and over 10 percent of the catalog had accuracy improved by more than 3.0 kilometers.
2. Average maneuver recovery time decreased by 6 hours.
3. Reduced sensor coverage gap times through scheduling efficiencies: average TVT was reduced by 78 minutes (across 98 percent of the catalog), and average RRI improved by 173 minutes on average (99 percent of the catalog).

The results indicate that SNARE holds the promise of improving SDA timeliness and information gain. The development and testing were encouraging enough for the sponsor to approve the development of a prototype to demonstrate the capability, connected to real sensors. To prove the ease of development and integration the project chose to deploy SNARE nodes on Intel NUCs, a \$300 64-bit Intel-based mini-PC with 32 GB of RAM. That single SNARE node was connected to a MITRE observatory telescope and was able to drive the sensor for testing. The second iteration was to connect three sensors into the network and build a front-end for functional demonstrations.

With consideration that SNARE has been shown to be a valid solution to the current resource allocation problem in comparison to the SP Tasker, the next sections will narrow in on both the SP Tasker algorithm and paradigm to give more complete context to these results. Then, SNARE will be evaluated on a deeper level in order to more directly compare the fundamental aspects of each.

### 3. CURRENT SP TASKER SENSOR SCHEDULING

The SSN currently utilizes a network of over 30 ground-based radars and optical telescopes in addition to on-orbit telescopes to gather information on over 20,000 RSOs. Managing a high quantity of objects given limited resources is a challenging problem that scales poorly as the catalog of objects increments. To maintain SDA, each object in the catalog requires a degree of certainty in which they must be positionally accounted for. While some objects are easily tracked by the network, others might be largely outside of coverage zones. While some objects are high-valued assets, others are debris from other missions. While some objects are known to be “friendly,” others are subject to scrutiny, and sometimes the intent or origin of an object is unknown or unclear. Each of these qualifiers leads to differences in both trackability and desired positional awareness, and these factors among many others make SDA very challenging to achieve, let alone sustain.

Today, there is no fully automated system in place for the SSN to do network-wide sensor tasking to help obtain certainty in the state of the catalog. The closest system utilized today by the SSN is the SP Tasker. The SP Tasker has been in use by the Space Defense Operations Center (SPADOC) since late 2005 as a replacement to the “greedy” tasker that was in place previously, which was so preferential to high-priority objects that it often disregarded objects of low priority when it sent out tasks. SPADOC determines priority levels for each object based on interest, desired positional accuracy levels, and other qualifiers. Although there is no ranking system that uniquely orders the importance of each object in the catalog, there is a system in place which assigns each object a Category (CAT) of 1-5 (which represents object priority) and Suffix of A-Z (which indicates collection requirements). An object that is a CAT 1A/2A is highly important and demands strict collection of information to satisfy requirements, whereas a CAT 3, 4, 5 with various suffixes will be of lesser priority which are tasked less often. Each combination of categories has different tasking requirements. Clearly, a CAT 1 or 2 will take tasking precedence, and will usually require as much information collection as possible from the network. The suffixes generally indicate how many tracks are required throughout the day, and that often varies depending on the type of sensor that will end up making the collection [1].

Information collection on an RSO comes in the form of a track of metric observations. Each observation in this track is the sensor measurement of the RSO (e.g., azimuth, elevation, range, and range rate) at a specific time. Correlated tracks are used to update the orbital state as specified in TLE updates. When tracks do not correlate to a known RSO then a new catalog entry is often made, and additional information must be gathered to improve confidence on its orbital state.

TLEs degrade in usefulness over time, and hence must be refreshed periodically to maintain positional awareness among the catalog; this is the functional intent of the SP Tasker. TLEs are refreshed based off daily observation collections, priorities of objects’ ebb and flow based off the measurements of recent collections, and the SP Tasker utilizes that information (among other relevant information about the catalog and the sensor network) to generate the Consolidated Task List (CTL) for the next day, attempting to maximize information gain between a dynamic sensor network with limited sensing capacities. In other words, the SP Tasker is trying to allocate work to a resource-limited network to satisfy the tasking requirements of the catalog with minimal loss of positional awareness [2].

The CTL that the SP Tasker generates is intentionally vague from a scheduling perspective. Essentially, each sensor is given a list of RSOs to collect on, where an expectation is set for how many observations and tracks should be

achieved. At a coarse level, this is a daily schedule although it lacks the specificity in timing of a more traditional scheduling problem. This is mainly due to the assumption that each sensor has some local process that can determine its own schedule to satisfy the tasks at hand. Part of what the SP Tasker attempts to encapsulate in the CTL is historical data on a sensor-by-sensor basis to determine which sensors can collect on which objects successfully, and approximately how many tracks a sensor can get per day based on empirical data.

#### 4. LIMITATIONS OF SP TASKER AND OPTIMIZED SCHEDULING

Aside from understanding what the sensor *can* do, the SP Tasker does not know what a sensor *will* do—hence why the output is not a rigid schedule. It is unreasonable for the SP Tasker to have a minute-by-minute understanding of what is needed by each individual sensor to achieve a collection. The full list of reasons may not be enumerable but ranges from things like how fast it takes for a sensor to physically move pointing from one point in the sky to another, to what the weather is like in the general area of the sensor’s coverage area. Not only do these types of factors depend on what kind of sensor is plugged into the network, but they also depend on factors external and dynamic to the network itself. This is the primary rationale for why the SP Tasker provides a coarse CTL over a rigid collection schedule.

Even if it were the case that the SP Tasker or another algorithm could design an effective, optimized, rigid schedule, there are some assumptions here that cannot be ignored. A traditional schedule is meant to be “fixed” to achieve optimality. In other words, every item on the schedule must occur precisely as scheduled to accomplish whatever objective is at play. When an item on the schedule falls through, takes longer than accounted for, or occurs in a way otherwise not planned for, the schedule itself may break down in validity. Further, it is impossible to account for unplanned events. What if a sensor experiences a temporary outage? What if an RSO starts maneuvering? What if something launches unexpectedly? Presumably, the calculus of what is important to collect information on rapidly changes in these situations. The SP Tasker is set up to run once per day, in which case these types of events are typically handled by manual schedule adjustments (by humans) throughout the day.

One could imagine setting up a system in which the SP Tasker, or another scheduling algorithm, is simply run every time the schedule does not play out as expected. However, there are still drawbacks to this. A collection scheme may only be optimized if the entire schedule is realized. Resolving the schedule may guarantee an optimal fixed solution moving forward but everything that has occurred prior is a component to a different solution—which means it may have only been optimal if the rest of the schedule happened as planned. Taking this to its logical extreme results in potentially extremely costly, and ultimately ineffective solutions—high risk, but low reward. Scheduling is best suited for mathematical environments in which everything is known about the system with high certainty. Scheduling employees if no one will be sick this week but accounting for planned time off, shift cycles and known busy periods may be an appropriate application for traditional scheduling optimization. Similarly, determining which cores in a computer to use for a set of tasks minimizing downtime is another ubiquitous example. In the first case, perhaps there is low risk/cost over time in needing to make alternate schedules due to unplanned events, and in the second case scheduling is predicated off strong, intimate knowledge of the problem space—in which case it can be carried out with high confidence with minimal risk and therefore cost. In either case, there is a clear cost-benefit to utilizing scheduling techniques and applying that to an extremely complex and dynamic system may not be the most appropriate approach.

#### 5. SNARE – A DECENTRALIZED SENSOR SELF-TASKING ALGORITHM

Although SNARE seeks to deliver many improvements over how SDA is done today, there are unique components of SNARE that must first be described and defined. SNARE in its entirety is more of an “environment” as opposed to an executable software routine run once per day. The main components of SNARE are as follows:

- Decision processes
- Collection processes
- Astrodynamics processes
- Data storage (also referred to as “data store”) and messaging processes

These processes are divided into tasks done by one of several types of nodes:

- Sensing nodes (sensors)
- Interfacing nodes (nodes which interface to the sensors)
- Validating nodes (nodes which validate and communicate data to other nodes)

If an interfacing node and a validating node exist together alongside a sensor, they may be referred to in aggregate as a “SNARE node.” Although this is not a comprehensive list of potential node configurations and processes, these are key to understand before understanding SNARE comprehensively. Each of these will be broken down individually, although a summary of the SNARE environment is provided first.

SNARE is composed of a set of sensors (referred to as sensing nodes), each of which communicates directly with interfacing nodes (i.e., a node that is paired with a sensing node). Interfacing nodes have access to the catalog of TLEs which are provided to the sensors to make collections (in the form of observations) as needed and as possible, per SNARE’s logic driver. Objects that have not been tracked recently or are suspect to require additional collections are given priority. Each sensor acts autonomously according to the same set of rules as every other sensor. Inevitably, an interfacing node will select appropriate work for the sensor. Once a process receives validation from the validating nodes, the work is attempted. If the collection results in observations, they are sent to the validating nodes to be checked for feasibility and are eventually processed using astrodynamics tools which update the state of the object(s) collected on in the form of a TLE. These TLEs, once validated, are then published for use by the network.

## 6. KEY NODES IN SNARE

An important feature of SNARE is sensor-agnosticism. This is not to say that different sensors cannot play different roles within SNARE, nor that sensor diversity is not an important consideration when adding sensors to the network. However, there are minimal requirements to be compatible sensors with the SNARE system. The basic requirement is that the sensor can collect and send information in the form of observations. The other requirement is that the sensing node in question must be able to communicate basically with an interfacing node. This will be covered again in a later section, but this essentially boils down to being able to ingest a priority list and determine which objects it can or cannot get observations on at a given time. This can be a simple process where the sensor makes a single collection according to the highest priority on the list or can be a higher-level process where some local optimizer determines from the priority list which set of objects can be tracked on simultaneously to maximize information gain. In any case, SNARE assumes that each sensor that is plugged into the network has personal knowledge of itself which supersedes in specificity what SNARE broadly knows about the sensors. Although SNARE will do basic propagation of objects to check for “feasibility” of collects, the sensors themselves are assumed to have the best understanding on what they can collect on and when. This eliminates inefficiencies that existing centralized tasking methods often have.

Each sensing node is accompanied by an interfacing node, as shown in Fig. 1. This node coordinates directly with the sensing node to implement the SNARE logic. The interfacing node is also the bridge between the actual collections within the network and the validating nodes. Simply put, the interfacing nodes help automate and organize the work done by the sensing nodes, whereas the validating nodes check feasibility of collections to make sure “reasonable” work is being performed, as well as check the network to make sure that unintentional duplication of work is not occurring. Although the sensing nodes act in autonomy in terms of choosing and timing collections, they are first approved by the validating nodes which make sure the collections are reasonable and unique among the rest of the collections that are occurring.

The validating nodes have more roles than just brokering collections within the network. It will be discussed a little more in detail in later sections, but the validating nodes are also responsible for performing the astrodynamics of SNARE. A basic propagation occurs to mitigate work redundancy, but further down the collection cycle (after the observations have been produced by the sensing node), the validating nodes will perform correlation checks on the observations to make sure the correct RSO is tagged in the information, update TLEs (or in special cases create new TLEs), as well as perform basic maneuver detection. Each of these processes are modular and are subject to improvements as developments on these processes occur over time. Once observations are validated, TLEs are produced. Once TLEs are produced then circumstantial information can be recorded, where applicable. All this information is then published for use by the other nodes in the network.

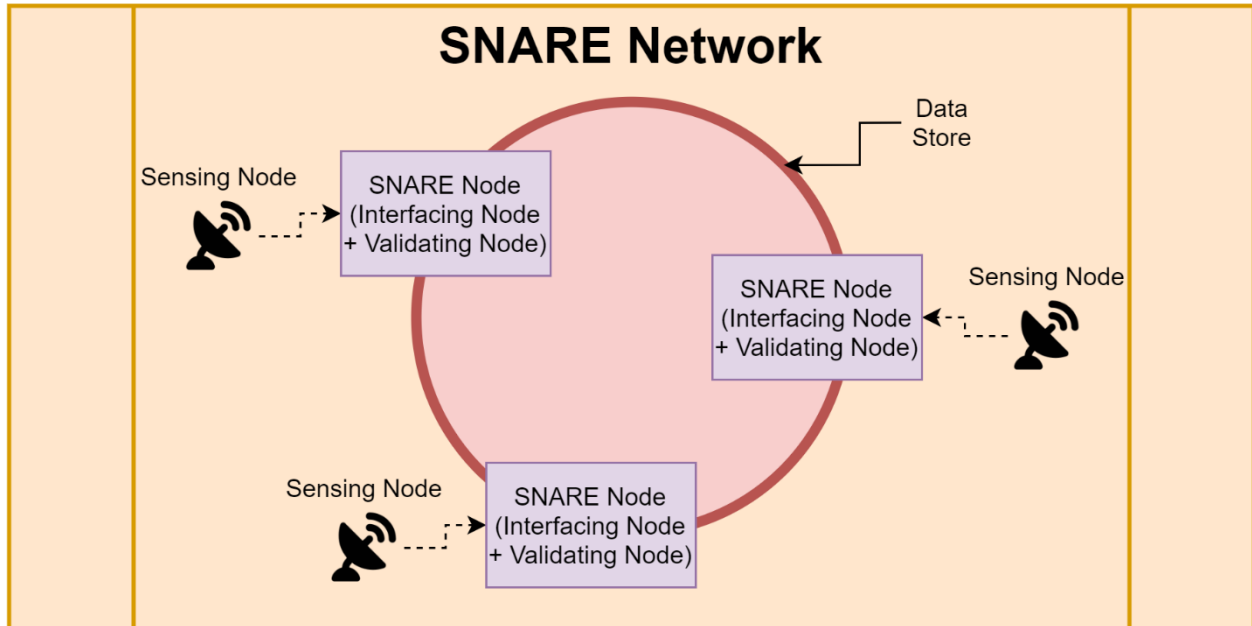


Fig.1. SNARE Network

## 7. SNARE LOGIC

Broadly speaking, SNARE logic operates fundamentally according to its Local Value Function (LVF). The LVF is the primary set of logic that exists within each interfacing node in the SNARE network. It is responsible for taking all the available information at hand and providing a framework by which each sensor can autonomously make collection decisions. Each interfacing node has the (same) most up-to-date positional knowledge of each object in the catalog. Currently, this positional state is derived from TLEs although there are other candidate options that could be explored in the future. Basic TLE propagation allows the interfacing node to narrow down a subset of the catalog to which the sensing node accompanying it can feasibly see. Once that is achieved, each object is rank ordered into a priority list. The direct output of the LVF is the “utility” of collect between a sensor and an object. The idea behind the utility measure is simple: given an object’s recent collection history and its collection requirements, how “useful” to the network is it to get a collection on it right now? This translates to a continuous function throughout time, to which every object has unique rates of value increase based on its individual collection requirements. Objects that are more important have stricter requirements meaning that their value of collection increases at a higher rate, which means that they are ultimately tasked at higher levels. It is important to note that requirements may be dynamic. Something that is perceived to be “space junk” one moment can have loose requirements, but if it suddenly maneuvers its requirements are automatically changed to reflect needing more information on the object. The system will react to this potential nefarious act by resolving more collections on it until stricter requirements are satisfied.

The current LVF is predicated by current tasking standards, which loosely boil down to revisit time (custody) and positional accuracy. Informed by requirements on these standards, each object has an associated “revisit expectation” (the desired maximum time between collections) and “observation expectation” (the desired minimum number of observations/tracks within recent history). Each of these expectations lead to real time continuous measurements throughout SNARE and are coupled with the expectation parameters to yield a deterministic value between zero and one for every object in the catalog throughout time. The numbers themselves do not have implicit meaning, but since information—and perhaps the requirements—for each object are unique, this produces a list of numbers for each object that can be used comparatively. Moreover, these values are a representative indicator of relative value at any given point in time. A sensor network that has an abundance of resources will see an average collect value across the catalog tend to zero, whereas a sensor network that is resource constrained will have an average catalog collect value that tends to one. This is a beneficial byproduct of the SNARE environment, because the “health” of the system’s collection abilities can be directly compared across time (assuming the requirements stay relatively the same in time).

The utility of collection by a given  $i$ th sensor in a sensing network for the  $k$ th object in a given catalog at time  $t$  is mathematically formulated as such:

$$U_{ik}(t) = L(v_{ik}(t), r_k(t), \hat{r}_k, o_k(t), \hat{o}_k) \quad (1)$$

where the utility of collecting on an object depends on the information available and some Local Value Function  $L$ . The required information is compiled in Table 1.

Table 1. A Summary of Local Value Function Components

Notation	Type	Description
$i$	Indicator	the $i$ th sensor
$k$	Indicator	the $k$ th RSO
$t$	Variable	time
$r_k(t)$	Variable	the time since the last validated track
$\hat{r}_k$	Parameter	maximum intended time between validated tracks
$o_k(t)$	Variable	the count of validated tracks within the last 24-hours
$\hat{o}_k$	Parameter	minimum intended count of validated tracks within the last 24-hours
$v_{ik}(t)$	Variable	indicates if a reasonable collection can occur

The key difference between SNARE logic and a scheduling process of any sort is that SNARE reacts to information that is known about the catalog at any given moment in time. A scheduler, as discussed earlier, cannot be reactive in the same way. Further, if SNARE can handle communication overhead appropriately, SNARE will utilize the true dynamic capacity of a sensor. A sensing node that is plugged into SNARE is either determining what its next collection should be or is carrying out a collection—there is minimal sensor “downtime.” In theory this could lead to greater sensor productivity, whereas today sensors are tasked based on empirical capacity estimations—estimations that are a direct product of how they have been tasked.

In summary, communal information about the RSO catalog is continuously read by the interfacing node to which collection values across the catalog are determined. As far as SNARE goes, two processes are always occurring in parallel: an interfacing node is reading new data and measuring value of collection between the RSO catalog, while its sensing node is making observations on the RSO catalog. The LVF provides a ranking of each object in the catalog in terms of what to collect on next, which is continuous throughout time and is the same between all sensors (if evaluated at the same time). The extent to which the value changes over time is a product of how strict the requirements are, but since the inputs to this logic are based off the most recent data set available about the catalog, changes in the system can be reacted upon quickly. The data that is communicated to the sensing node as an outcome of the LVF is just a rank-ordered priority list of RSOs. The sensing node then locally optimizes collection based on that list. If a sensing node can only do one task at a time, then it would take the top feasible collection on the list to collect upon (this single task is then validated by the validating nodes). If the sensing node can multi-task or is actually a collection of smaller and independent sensing nodes, then it is up to the sensing node itself to locally optimize feasible next-collections based on the priority list (in which case the set of tasks is evaluated for feasibility by the validating nodes). Looking at “next-only” tasks for a sensor can lead to higher sensor utilization overall, which means the sensors will be outputting more data than in traditional tasking settings. The output of the collection by the sensing node is in the form of observations, which are sent back to the validating nodes via the accompanied interfacing node.

## 8. COLLECTION PROCESSING

Once a sensing node chooses work, gets validation and approval from the validating nodes, and then completes a collection, the observations are communicated to the accompanying interfacing node. The interfacing node then submits the completed observations to the validating nodes, where processing can begin. The process here is like the processing that occurs within the SSN today, but automated, and occurs “real-time” as opposed to once per shift-cycles or once per day. First, it is important to recognize that observations are accepted from a sensing node after validation occurs. If there is no initial validation, there can be no observations. Assuming validation occurs, the object is checked via a correlation process to see if the RSO a sensing node attempted to get data on is probably the object

in the data collected. In certain cases, it is possible that a different object was collected on by accident. This data is still useful to the SNARE network, so it may be kept—assuming the validating nodes reach consensus on what the object is. Those observations where uncertainty is high for which object it is may be sent to an orbit determination process, which creates a temporary new satellite catalog number in the system, to which the default collection requirements are usually high (the idea in SNARE being that when uncertainty is high you need more information to reduce future uncertainty). If the uncertainty is relatively low, then the new information (along with relevant recent history) is sent to an orbit update process. Here, the TLE is refreshed and most accurately reflects the positional state of the object. Presumably, certain TLE update processes also contain information on how “good” the update is. Again, if uncertainty is high then the requirements for the object can be modified to influence future collections. If a TLE is updated for an object, then a maneuver determination algorithm is used—again, to check the uncertainty level of the object. Presently, SNARE utilizes the Time Off Element Set (TOES) calculation to inform maneuvers. If an object is determined to (potentially) having recently maneuvered, the value of collection for it skyrockets until custody is maintained. In the case where many objects maneuver simultaneously, their previous ranking and requirements are sometimes used for the LVF to determine what has priority tasking if there are duplicate ranking values. If each of these processes reach a consensus among the validating nodes, the updated requirements and new information are stored and made available for each interfacing node in the SNARE network.

## 9. DECENTRALIZATION AND DATA APPROACH

The approach for design and implementation of the SNARE data store is driven by the degrees of centralization and resilience required by the capability. Both centralization and resilience need to be considered over the life cycle of the SNARE network as a capability. SNARE is partially decoupled from CSpOC and operates in a relatively decentralized manner, responsible for its own tasking, collecting, and first order analysis such as TOES. In contrast, present day CSpOC uses the SP Tasker daily regimen which centralizes tasking, sending specific collecting requirements to applicable sensors. Internal to SNARE, tasking is further decentralized to be the responsibility of the SNARE node assigned to each individual sensor. As described above, this decentralized behavior requires a data store approach that can provide a common set of data across all SNARE nodes. Note that there is no particular SNARE node which assumes responsibility for the data store on behalf of all the other SNARE nodes.

### Resilience

Capabilities are built and used for specific purposes, and the degree of resilience a capability possesses can span hardware, software, workforce, mechanical and environmental factors. This subsection focuses on the resilience of the SNARE data store to enable SNARE to resist, recover, and continue after accident, attack, or attempted deceptive takeover. Data resilience can best be described by illustrating scenarios of accident, attack, and deceptive takeover. For example:

- Resilience Scenario 1 – loss of a SNARE sensor. The SNARE model accommodates both dynamically adding and removing sensors, without impacting the rest of the network. The reason is that the coordination is accomplished indirectly with the data layer.
- Resilience Scenario 2 – network attack separates CSpOC from SNARE. The SNARE network continues to task sensors and collect observations. CSpOC connects once connectivity is restored, and the SNARE data node that CSpOC uses then syncs with the rest of the network, and CSpOC does not lose data.
- Resilience Scenario 3 – admin accidentally overwrites SNARE node data. In this, if using decentralized technology such as blockchain, the tampering is immediately detected, the impacted data node is ignored, can be removed, repaired, and put back online.

### Decentralization

As discussed above, the SNARE network is decentralized from CSpOC for the purpose of sensor tasking and collections. However, SNARE remains under administrative, operational, and tactical control of the CSpOC. This subsection considers further decentralization within SNARE, driven by possibilities of enhanced sensor partnerships to improve timeliness and quality of sensor data.

For purposes of the initial operational prototype, the SNARE sensors are the dedicated sensors in the SSN. The pedigree is known, and the sensors (and supporting organization) are trusted by CSpOC. Thus, the control of nodes and associated data within SNARE is singular, not decentralized. However, there are possible scenarios for future SNARE iterations which would drive decentralization of control of data within SNARE. Also, the SNARE data is processed by CSpOC then published as needed. If sensor partnerships are created to improve the effectiveness of SNARE, this may drive decentralization of the control of data within SNARE. For example:

- Decentralization Scenario 1 – add sensor data (but not integrate a sensor) to SNARE. For example, CSpOC may want to inject a TLE from an external sensor to drive future tasking and collection.
- Decentralization Scenario 2 – integrate partner nation sensor. This is like scenario 1, except instead of one-time injects, the injects are continuous.
- Decentralization Scenario 3 – integrate open-stare and/or other related sensor capabilities.
- Decentralization Scenario 4 – integrate partner nation sensor directly in SNARE. This is tighter integration than other scenarios. In this case, the SNARE network can not only ingest the partner sensor data, but also drive the tasking of the partner nation sensor.

### **Data Store Requirements for Operational Resilience and Decentralization**

The data store requirements, required to support the resilience and decentralization scenarios above are stated below. If scenarios are relaxed, then data store requirements may also be relaxed however, the full tradespace of combinations of scenarios and resulting data store requirements is beyond the scope of this paper. The following data store requirements support the resilience and decentralization scenarios:

- Posted collected data must be provably attributed to the applicable sensor.
- Posted first order processing of data must be provably attributed to the applicable validating node.
- Data must be tamper-evident and redundantly available across the SNARE network.
- Loss of data from any single node must not prevent SNARE from operating.
- Loss of connection between a SNARE node and the rest of SNARE must not prevent either the isolated node nor the balance of the SNARE network from operating; the isolated SNARE node and the balance of SNARE network can be repatriated and the isolated SNARE node data (collections, first order processing) can be incorporated into the SNARE network.
- Loss of connection between SNARE and CSpOC must not prevent SNARE from operating.

### **SNARE Data Store Technical Approaches**

The data store requirements above, inform the tradespace of technical approaches for the SNARE data store. There are several approaches for how SNARE can use a data store to coordinate sensor tasking and collection. In this section, the standard centralized database (DB) is compared to two different decentralized data methods: Distributed Ledger Technology (DLT), and permissioned blockchain.

Decentralizing the current CSpOC paradigm into a network of sensors, whose associated nodes can perform tasking and collection in parallel, shortens processing times which drives more efficient collections. However, decentralization requires data that is trusted among all sensors, since each collection processed informs the following collection. Trust among SNARE sensors drives the data store technical approach. Trust is affected by earlier scenario choices of which sensors (e.g., open stare, partner nation) are included in the SNARE network beyond the initial set of dedicated SSN sensors.

#### Centralized Data Environment

A centralized DB is simplest. However, it comes with risk of a single point of failure (e.g., accident or attack), including edits to data or even destruction of data (insider or malicious). Therefore, the trust among sensors and users may be lower than that of DLT or blockchain, discussed next. Centralized data store approach is the most common and occurs where a single organization owns and operates the data store. However, complex missions such as the resilience and decentralization scenarios above describe, may require non-traditional participants which may reduce trust in this data store model.

## Distributed Ledger Technology (DLT)

DLT databases use data replication to mitigate the risk of single point of failure inherent in centralized databases. The replication is coordinated with its transaction log (all DBs have a transaction log). The log is the basis for keeping all the distributed database replicas up to date on the data set. Having redundant copies on the network can increase performance when there are many read/query requests on the data. It also increases the resilience of the overall distributed database by having other replicas available to respond to requests in the event of a machine crashing (accident or attack). However, a DLT requires central control, which may be incompatible with some of the resilience and decentralization scenarios.

## Blockchain

Blockchain shares some characterizations of a DLT, such as distributing and replicating data across nodes. However, blockchain goes a step further and provides an immutable record of transactions. However, to provide such an immutable store, the blockchain model is stripped down to just a transaction log and does not provide higher order data store functions such as tables, views, or advanced queries. The immutable record of transactions is supported by unique cryptographic capabilities to ensure the integrity of the transaction log against tampering [3] [4]. As a result, blockchain data stores replicate data across all nodes on the network and can withstand traditional faults well as potential malicious activity. Thus, blockchain enables a distributed, replicated, log of transactions and is well-suited for sharing among diverse stakeholders.

## Tradespace

The primary difference between the data store approaches above is degree of centralized implementation and control over the data store. In a centralized database, one organization owns and controls the database from a centralized location. In a DLT, data nodes may be distributed however, even though the data is distributed (redundancy), all nodes are owned and operated by one organization. While the redundant nodes provide resilience to accidents, the nodes are more susceptible to attack including insider threat due to the centralization of control. In the fully decentralized blockchain, the distributed nodes may be owned and operated by different parties, which provides resilience to both accident and attack.

The technology is most mature for centralized, and most rapidly evolving for decentralized permissioned blockchain, with DLT also evolving. As a result, the centralized database has a readily available workforce for projects, while blockchain projects will need to consider workforce shaping to execute development tasks.

## **10. BEYOND SNARE**

SNARE uses decentralized data to enable functional coordination required to perform decentralized sensor tasking and collection. Potentially there are additional areas in the space domain where decentralized data can enable functional coordination. For example, BESTA (Blockchain Enabled Space Traffic Awareness) and GREAT (Global Resilient Extensible Autonomous Trusted) are MITRE research projects, in collaboration with space subject matter experts, which explore use cases for broader international space information sharing [5].

BESTA and GREAT look beyond improved SDA provided by SNARE, to data pertaining to operational decision making, such as intent to maneuver. Presently, there is no requirement to share intent to maneuver, and even if there was a requirement to share, there is no means. Sharing in the international arena requires decentralized information sharing to assure no one party has control over the information flow, that all information posted is attributed, and that provenance and pedigree of the information is known so that the reader of the information can decide how best to use it.

Pedigree and provenance (sometimes called traceability) are being vigorously pursued in the manufacturing supply chain realm where critical functions need to assure that they are using genuine parts. For example, a nuclear power plant operator must assure the industrial controls used to manage fuel rods have genuine microelectronics, mechanical actuators, and embedded software. Like manufacturing supply chains, space traffic operations can benefit from

additional maneuver information, where that information must be traceable and trusted. In effect, trusted information which informs decision making contributes to a space operations decision making supply chain. If viewed in this way, space operations decision making may benefit from improvements in manufacturing supply chain traceability.

Some similarities between manufacturing supply chain traceability and space operations decision making regarding the constraint of bilateral information exchanges are striking:

- Present information sharing in both occurs through bilateral, stakeholder-to-stakeholder data exchanges. For example, early in 2021 NASA and SpaceX entered a bilateral information sharing arrangement for intent to maneuver and other data.
- In manufacturing supply chains, traceability across tiers (bilateral connections) is a major concern. Please see NIST SP 800-161 Supply Chain Risk Management Practices for Federal Information Systems and Organizations [6] for a full description of the problem. Revision 1 is in progress and taking comments.
- Note that a recent study from Laval University has identified an “explosion” in bilateral agreements since 2010 and remarkably, that most of the 931 agreements they have identified do not include a party, either public or private, from the United States [7].

Opacity of information is one of the biggest risks in manufacturing supply chain and is one of the biggest risks in space traffic operations. Both may be addressed by sharing information wider than bilaterally, where this requires trust and may be implemented using decentralized information sharing technology such as blockchain or DLT.

## 11. CONCLUSION

SNARE holds the promise of improving tactical relevance of SDA, by providing near real time positional state of objects in the space domain for timely, accurate, and actionable. SNARE has been transferred to USSF and operational prototype phase is underway. Future papers will describe subsequent improvements and results.

Other sensor networks may also benefit from a decentralization approach. However, each situation is different, and the authors recommend careful analysis, modeling, prototyping, and testing for each situation to generate evidence-based results and avoid blanket generalizations. Nonetheless, the authors anticipate there will be additional opportunities for SNARE-like decentralized sensor tasking and collection beyond SDA.

## 12. REFERENCES

- [1] J. G. Miller, “A new sensor allocation algorithm for the space surveillance network,” 74th MORS Symposium, vol. 5, August 2006.
- [2] North American Aerospace Defense Command Cheyenne Mountain Complex Integrated Tactical Warning and Attack Assessment Program Technical Report-Study Services Special Perturbations Tasker Programmer Documentation for the Integrated Space Command and Control (ISC2), Lockheed Martin Information Systems and Global Solutions, 9970 Federal Drive Colorado Springs, CO 80921-3616, July 2014.
- [3] D. Bryson, D. Penny, D. Goldenberg, and G. Serrao, “Blockchain technology for government,” MITRE, MTR180046, December 2017.
- [4] D. Bryson, G. Serrao, and H. Reed, “Blockchain for decentralized missions: Technical report for chief engineers and technologists,” MITRE, MTR190397, September 2019.
- [5] H. Reed, N. Daily, R. Carden, D. Bryson, “Blockchain Enabled Space Traffic Awareness (BESTA): Discovery of anomalous behavior supporting automated space traffic management,” AIAA ASCEND 2020, virtual, November 2020.

[6] NIST SP 800-161 Supply Chain Risk Management Practices for Federal Information Systems and Organizations, [NIST Releases Draft of NIST SP 800-161, Revision 1 for comment, Cyber Supply Chain Risk Management Practices for Systems and Organizations. | NIST](#)

[7] J. Morin, E. Tepper, "Mapping the Polycentric Governance of Space" (video report), University of Laval, Quebec Canada, URL: <https://www.youtube.com/watch?v=OEbACdfR9wU> (last accessed August 2021).

### 13. ACRONYMS

AMOS	Advanced Maui Optical and Space Surveillance Technologies (conference)
BESTA	Blockchain Enabled Space Traffic Awareness
CAT	category
CSpOC	Combined Space Operations Center
CTL	Consolidated Task List
DB	database
DLT	Distributed Ledger Technology
FY	fiscal year
GEO	Geosynchronous Earth Orbit
GPS	Global Positioning System
GREAT	Global Resilient Extensible Autonomous Trusted (systems)
GSW	Global Sensor Watch
ISC2	Integrated Space Command and Control
LEO	Low Earth Orbit
LVF	Local Value Function
NASA	National Aeronautics and Space Administration
NIST	National Institute of Standards and Technology
NP	Non-deterministic Polynomial-time
PNT	Positioning, Navigation, and Timing
RRI	Revisit Rate Interval
RSO	resident space object
SDA	Space Domain Awareness
SID	Satellite Identification
SMC/SPG	Space and Missile Systems Center Space Domain Awareness Division
SNARE	Sensor Network, Autonomous, Resilient, Extensible
SP TASKER	Special Perturbations Tasker
SPACECOM	Space Command
SPADOC	Space Defense Operations Center
SSN	Space Surveillance Network
TLE	Two-Line Element
TOES	Time Off Element Set
TVT	Total Violation Time
USSF	United States Space Force