NASA/TM–20240012189

Discovery and Synchronization Service Architectural Notes

Joseph L. Rios
Ames Research Center, Moffett Field, CA

September 2024

NASA STI Program Report Series

The NASA STI Program collects, organizes, provides for archiving, and disseminates NASA's STI. The NASA STI program provides access to the NTRS Registered and its public interface, the NASA Technical Reports Server, thus providing one of the largest collections of aeronautical and space science STI in the world. Results are published in both non-NASA channels and by NASA in the NASA STI Report Series, which includes the following report types:

- TECHNICAL PUBLICATION. Reports of completed research or a major significant phase of research that present the results of NASA Programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA counterpart of peer-reviewed formal professional papers but has less stringent limitations on manuscript length and extent of graphic presentations.

- TECHNICAL MEMORANDUM. Scientific and technical findings that are preliminary or of specialized interest, e.g., quick release reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.

- CONTRACTOR REPORT. Scientific and technical findings by NASA-sponsored contractors and grantees.

- CONFERENCE PUBLICATION. Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or co-sponsored by NASA.

- SPECIAL PUBLICATION. Scientific, technical, or historical information from NASA programs, projects, and missions, often concerned with subjects having substantial public interest.

- TECHNICAL TRANSLATION. English-language translations of foreign scientific and technical material pertinent to NASA's mission.

Specialized services also include organizing and publishing research results, distributing specialized research announcements and feeds, providing information desk and personal search support, and enabling data exchange services.

For more information about the NASA STI program, see the following:

- Access the NASA STI program home page at http://www.sti.nasa.gov

- Help desk contact information:

https://www.sti.nasa.gov/sti-contact-form/
and select the "General" help request type.

Acknowledgments

The Discovery and Synchronization Service emerged as a response from industry to a need initially identified by the NASA UTM Project. Significant work from industry went into the definition and implementation of DSS. The InterUSS Platform as a project of the Linux Foundation deserves a great deal of credit for that implementation and continuing maintenance and maturation of DSS. The ASTM working groups that developed the formal requirements for DSS also need to be thanked for their significant work in this area. The FAA subsequently being open to the use of DSS as part of an operational UTM system deserves kudos for this openness to allow industry to proffer and demonstrate workable solutions. Finally, thanks to the NASA team including Irene Smith, Punam Verma, Charlene Cayabyab, Stefan Shuet, and Dan Liddell for careful review of this short document.

Document Version	**Date**	**Changes**
Original	27 Sept 2024	-
Errata	16 Oct 2024	Remove extra title page. Add page numbers. Add definitions of Database Node and Database Cluster to Table 1 for clarity.

This report is available in electronic form at https://sti.nasa.gov

"I do not seek. I find."

— **Pablo Picasso**

Overview

This is a short architectural note for a critical component of safely scaling drone operations. The primary motivation for this note is to collect language and terminology around the Discovery and Synchronization Service (DSS) as it is currently implemented for Uncrewed Aircraft System (UAS) Traffic Management.

The major driver for this document was frequent miscommunication about DSS. Often there have been multiple informed and invested stakeholders in the same room making statements about "multiple DSSs" and each of them meaning something slightly different.

This document will NOT:

- describe or justify UTM
- discuss the functionality, development, deployment, or maintenance of DSS
- cover authentication or authorization topics
- summarize any standards
- outline the governance around DSS or UTM services

To understand UTM, the interested reader is directed to the Federal Aviation Administration (FAA) website[1] that contains updated information and links to important UTM documentation. To learn more about the DSS implementation (potentially useful before and/or after reading this note), the InterUSS repositories[2] are full of important technical information, including data schemas, sequence diagrams[3], testing information[4], and related documentation. As of this writing, two ASTM standards[5,6] define requirements related to DSS and those are helpful references to understand overall drivers for the current implementation.

To take full advantage of this documentation process, a brief discussion of other DSS architectural aspects is also provided after the Terminology and Relationships sections.

---

[1] https://www.faa.gov/uas/advanced_operations/traffic_management, accessed 13 Sept 2024.
[2] https://github.com/interuss, accessed 13 Sept 2024.
[3] https://github.com/interuss/dss/blob/master/concepts.md, accessed 18 Sept 2024, describes DSS interactions via sequence diagrams quite well.
[4] https://github.com/interuss/monitoring/blob/main/monitoring/README.md, accessed 18 Sept 2024, discusses testing of USS and DSS implementations.
[5] https://www.astm.org/f3411-22a.html, accessed 17 Sept 2024.
[6] https://www.astm.org/f3548-21.html, accessed 17 Sept 2024.

Terminology

For fruitful discussions about complex technology, precise terminology is critical. In Table 1, we provide the key DSS terms and their definitions. Where possible, a source for the definition is provided. Note that this document uses capitals for defined terms, while ASTM and InterUSS capitalize just the first word in a term. For example, the source material uses “DSS instance” where this document uses “DSS Instance.” Sometimes a definition exists for a given term, but it is redefined here to better capture its essence or as-built qualities.

Table 1. Definitions of key terms related to the DSS, presented alphabetically.

Term	Definition	Source
CockroachDB	CockroachDB is a distributed SQL database built on a transactional and strongly consistent key-value store. *[The current DSS persistence layer]*	CRDB FAQ
CRDB	CockroachDB	
CRDB Cluster	A group of interconnected CockroachDB nodes that function as a single distributed SQL database server. Nodes collaboratively organize transactions and rebalance workload and data storage to optimize performance and fault-tolerance.	CRDB Architecture
CRDB Node	An individual instance of CRDB.	CRDB Architecture
DAR	DSS Airspace Representation	
Database Cluster	A collection of Database Nodes working together to function as a single, logical database.	This document
Database Node	An individual instance of a database, typically deployed on its own server.	This document
Discovery	The process of determining the set of USSs with which data exchange is required for some UTM function.	ASTM F3548-21
Discovery and Synchronization Service	A service defined in [F3548-21] that enables USSs to discover other USSs with which data exchange is required and to ensure that USSs use current and consistent entity data.	ASTM F3548-21
DSS	Discovery and Synchronization Service	
DSS Airspace Representation	A single, consistent representation of all entity references and subscriptions in the airspace of a DSS pool and provides access to those entity references and subscriptions on the basis of an area and time of interest.	ASTM F3548-21
DSS API Service	A service that implements endpoints that allow USS clients to access the DSS Airspace Representation.	This document
DSS Instance	A deployment by a single organization of CRDB Nodes, a DSS API Service, and the connections between them meeting DSS requirements in support of a DSS Region.	This document
DSS Instance	For availability purposes, multiple synchronized copies of the DSS supporting a DSS region. Each copy is referred to as a DSS instance.	ASTM F3548-21
DSS Pool	A synchronized set of DSS instances where operations may be performed on any instance with the same result, and information may be queried from any instance with the same result.	ASTM F3548-21
DSS Region	The geographic area supported by a DSS pool.	ASTM F3548-21
Entity	[In the context of DSS,] a generic term referring to types of data that need to be shared between USSs.	ASTM F3548-21
Reference Implementation	The implementation of a standard to be used as a definitive interpretation for the requirements in that standard. Reference implementations can serve many purposes. They can be used to verify that the standard is implementable, validate conformance test tools, and support interoperability testing among other implementations. A reference implementation may or may not have the quality of a commercial product or service that implements the standard.	NISTIR 8074 Volume 2 NIST Glossary

Relationships

The following diagram illustrates some of the components of the DSS reference implementation and the relationships of those components with some of the terms and concepts from other source documents.

Relationship Diagram

This diagram is "UML-like" as it is not strict UML. Due to the divergence from the UML standard, a legend is provided.

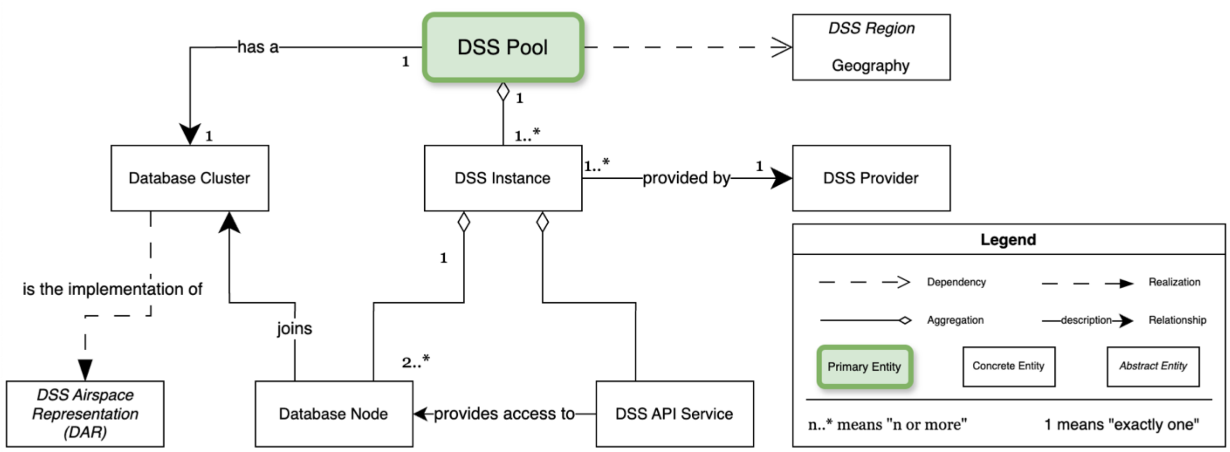

Figure 1. Diagram illustrating relationships between DSS elements.

Relationship Table

The following table describes each pairwise relationship with a minimal sentence.

Table 2. Description of the pairwise relationships between DSS elements.

Element A	Element B	Relationship
DSS Pool	Database Cluster	A DSS Pool has a Database Cluster.
DSS Pool	DSS Instance	A DSS Pool is composed of one or more DSS Instances.
DSS Instance	Database Node	A DSS Instance has two or more Database Nodes.
DSS Instance	DSS API Service	A DSS Instance has a DSS API Service.
DSS API Service	Database Node	A DSS API Service provides access to Database Nodes.
DSS Provider	DSS Instance	A DSS Provider deploys and maintains a DSS Instance.
Database Node	Database Cluster	Each Database Node is a part of one Database Cluster.
Database Cluster	DAR	A Database Cluster is the implementation of DAR.

DSS Architectural Sidenotes

Persistence Layer

The current iteration of DSS leverages CockroachDB (CRDB) as its persistence layer. The architecture allows for another technology choice for the persistence layer without compromising the overall structure defined in Figure 1. A different choice may impact other elements in terms of the interfaces and implementation, but the system would look like what is presented here. For example, a “cluster” for a Relational Database Management System (e.g. Postgres or MySQL) might mean replicated nodes with a primary and multiple secondary nodes as opposed to a truly distributed datastore like CRDB (or any similar technology) provides.

Future Services

While DSS is a system with requirements within two international standards (as of this writing), there is value in widening the aperture on DSS beyond these existing standards. For future services that have a requirement to “discover” things and to ensure that data are “synchronized” across independent systems, there is a need for a Discovery Service and for Synchronization. Technically these functions do not have to live with each other, but practically speaking strong operational value comes from having them within the same service, namely DSS.

For future services a few questions should drive discussion of the discovery. Specifically:

1. What is the resource that requires some sort of arbitration amongst federated systems?
2. What needs to be discovered about the resource?
3. What qualities are most important[7] to emphasize in the discovery system?

For Strategic Conflict Detection in UTM, the answers could be the following:

1. Volumes of airspace, represented as 2D grid cells.
2. Operations in the cells and how to contact the service providers for those operations.
3. Consistency, Survivability, Interoperability.

With those answers, one might see how the choice of CRDB came to be and the overall set of relationships (Figure 1 and Table 2) between systems eventuated.

For another future service, say a vertiport management system for larger vehicles or a high-altitude constraint service, the answers to the three questions above may be different and may very well lead to a different system definition.

[7] When designing or architecting a system, it is easy for stakeholders to want many qualities (modifiability, maintainability, availability, integrability, etc.) to be maximized at the same time. However, often these qualities compete against each other or imply different design solutions. Thus, it is best practice to determine the most important qualities via a stakeholder engagement process. Something like 3 or 4 qualities are good to focus on initially.

Even if the answers for those questions are the same for a new service as they are for Strategic Conflict Detection, the follow-on design decisions could lead to CRDB or a different persistence mechanism. It could be the same primary components connected as described in Figure 1, but with a new DSS API Service. Proper architectural analysis must be performed to get to a reasonable DSS for the service at hand.

As a historical note, the current DSS implementation and requirements were partly driven by the stakeholder goal of allowing for multiple organizations to collaboratively host DSS so that it had some level of robustness and avoided the need to “pick a winner” to be the operational DSS host.

DSS Reuse

It may be necessary or just desired that the same DSS Pool handles discovery for multiple services. In such cases, careful consideration of the DAR is important. Do the services share a DAR? If so, is the implementation of the DAR appropriate for each of the services? In addition, stakeholder discussion related to the desired qualities of the system are aligned for each service. Tradeoffs for availability, consistency, and latency for example are a well-documented[8] concern in distributed databases. So, if the DSS was to support discovery for three different services and those services each emphasized a separate one of those three qualities, then the system designer will be in serious trouble.

In the case where the DAR and the qualities of the system all align, maybe separate DSS API Services or an updated DSS API Service will meet the need. Alternatively, if separate DARs are needed and even separate persistence mechanisms are required, an updated DSS API Service may hide that complexity from clients. There are several ways to work the architecture to meet the needs of stakeholders.

Multiple DSS Pools

It is possible that multiple DSS Pools are needed to meet the needs of stakeholders. In this case, careful consideration of the interoperability between DSS Pools and the appropriate authorization for clients to access the various DSS Pools is required. This issue has been examined previously in a limited manner, but as Concepts of Operations mature for such scenarios, the issue will need to be addressed more systematically.

If two DSS Pools need to interoperate, it is much like two inmates playing chess in different cells, calling out moves to one another and tracking the game on different boards. This can work but is also error prone.

[8] D. Abadi, "Consistency Tradeoffs in Modern Distributed Database System Design: CAP is Only Part of the Story," in *Computer*, vol. 45, no. 2, pp. 37-42, Feb. 2012, doi: 10.1109/MC.2012.33.