SST DATA TRUST AS A SERVICE - TOWARDS SECURE DE-CENTRALISED MANAGEMENT AND EXCHANGE OF SPACE SURVEILLANCE AND TRACKING DATA

M. Popp[(1)], V. Rogojin[(1)], M.C.Boysan[(1)], and M. Wallum[(2)]

[(1)]*Guardtime, Tallinn, Estonia, Email: {Marika.Popp,Vladimir.Rogojin,Mehmet.Boysan}@guardtime.com*
[(2)]*European Space Agency, Email: Marcus.Wallum@esa.int*

ABSTRACT

Near-Earth orbital space becomes increasingly complex in terms of the number of assets and actors, hence requiring a system that monitors space and providing real-time, high quality and actionable situational awareness to all stakeholders. The main challenges for space actors to operate in a confident and error-free manner, agree on responsibilities and attributions while ensuring openness, security and sustainability of space operations are related to establishing integrity, availability and a clear consensus on Space Surveillance and Tracking (SST) data from multiple authenticated data sources. We propose a SST Data Trust as a Service solution prototype, based on Distributed Ledger Technologies (Blockchain) and implemented within Hyperledger Fabric framework. The article demonstrates that a blockchain-based solution naturally fits the role of a decentralized, secure and trusted SST data sharing platform, resulting in a framework not controlled by any single entity, yet acting as a single system, synchronised across all participants and application areas.

Keywords: Blockchain; DLT; Trustless de-centralized framework; Hyperledger Fabric; Space Survelance and Tracking; Collision Risk Estmation and automated Mitigation.

1. INTRODUCTION

The application of DLT to the space domain is an emerging area of research, with studies ranging from application to corporate functions such as procurement and information management, security in distributed space inter-networks and trust among multiple operators/control centers, distributed mission operations and porting on-board software to smart contracts, decentralized access control for Space Situational Awareness systems and the use of blockchain in multi-sensor satellite architectures. However, a dedicated study on the possible application of the technology to the ESA mission operations domain and specifically to the ground segment had not yet been conducted. Hereby, the paper presents our studies and first Proof-of-Concept regarding the subject.

DLT for *Space Situational Awareness* (SSA) was selected as the use case for prototype demonstration due to several reasons. Today, the growing emergence of large constellations and increasing launch rates is driving the global demand for improved, automated SSA services. Moreover, a trend towards increasing numbers of *Space Surveillance and Tracking* (SST) data and service providers, including commercial actors, increases the overall complexity of the ecosystem and emphasises the need for secure, trusted (consensus-based) mechanisms to ensure the reliability and trust for SST data consumers and other stakeholders. Immutability and provenance of the increasing volumes of SST data is of key importance since the data is being shared to produce collision warnings and other critical surveillance and tracking products. Yet today there is no open, widely trusted, global source of SST data accessible for global users across multiple application domains. Instead, various ground and space-based systems are being operated, mostly of governmental origin and with implicit trust of the data provider.

The paper addresses the challenges outlined above by proposing a solution which would allow space actors to operate confidently, avoiding human error, agreeing on responsibilities and attributions, and ensuring that space operations remain open, secure, and sustainable. This can only be achieved by developing tools for ensuring integrity, availability, and a clear consensus on SST data from multiple authenticated data sources. Based on such consensus, this data may be then used with confidence, for example for automating planning and maneuver decisions based on shared rules/operations concepts across multiple operators in support of a reliable Space Traffic Control capability.

Based on thorough research, a prototype solution of SD-TaaS was developed to enable SST Catalog Data providing entities to share data with each other in a decentralized manner, with full confidence and trust that the integrity and provenance of the shared data is conserved. For this, a secure distributed ledger (blockchain) for SST data has been implemented, to ensure integrity, add resiliency, and allow for users to reach consensus on shared SST data across SST data catalogs.

The SDTaaS prototype system is represented by business logic, web, and client separate application tiers. The solution implements the prototype within Hyperledger Fabric framework [8, 12] that naturally allows to host and operate multiple separate tier-specific peers by all participants. In this manner, the proposed framework includes high flexibility in choosing and implementing future administrative, trust and computation distribution schemes among all the participants.

2. SPACE SITUATIONAL AWARENESS

For decades, mainly the United States has been dominating the SSA domain worldwide. The adoption of the Space Policy Directive-3 in 2018 for National Space Traffic Management (STM) Policy [15] was an important landmark in this. There is yet no comparable STM policy developed in Europe. According to ESPI, the gap between European and American capabilities creates a situation of reliance/dependence for European stakeholders and an imbalance in cooperative arrangements. Such reliance on U.S. capabilities presents some limitations related to its strategic implications. [3]. Hence both the EU and ESA have been increasingly active in investigating further opportunities for improving European SST capabilities for which secure exchange of SST data is a critical cornerstone.

One of the goals of the U.S. STM Policy is "to improve SSA data interoperability and enable greater SSA data sharing". This is also the case for European entities and can be seen as posing significant challenges but also many opportunities since relying on multiple data sources, as planned for the U.S. open architecture data repository, will raise new interoperability challenges to ensure data quality, integrity, availability and confidentiality. "Even though the U.S. already has far better capabilities today, Europe has much to offer to an open architecture data repository." [3].

Recent developments of the U.S. space strategy have also supported the growth of the countrys SST commercial sector. TruSat [1] is a U.S. initiative led by ConsenSys Space in partnership with the Secure World Foundation, the Society of Women in Space Exploration and Moriba Jah, a space scientist and aerospace engineer at the University of Texas at Austin. It is aimed at analyzing the naked-eye satellite observations that are made by volunteers and submitted via the app, to come up with more accurate information about the orbits of thousands of satellites. Blockchain technology would be used in this case to provide transparency about the source of orbital data. Similarly, U.S. based L3Harris is aimed at maintaining and upgrading radar and optical sensors and command and control systems that provide timely, accurate space domain awareness data for military, civil and commercial users. L3Harris has also been involved in working on an end-to-end framework for decentralized Space Domain Awareness (SDA) that enables worldwide Space Traffic Management (STM), where in the interest of tracking potential threats to on-orbit space systems, all nations and stakeholders undergo the open, immutable, and transparent exchange of SDA data in [18]. Similarily, attempts to support decentralizing the space mission by creating a highly available, secure, trusted data layer of SSA and agreement data have been described in [16] which further proves to underline the growing importance of secure and trusted SST data exchange networks for globally networked future space operations.

ESA's Space Safety Programme that kicked-off in 2020 includes as a cornerstone the *Collision Risk Estimation and Automated Mitigation* (CREAM). CREAM entails the development of technologies for automating collision avoidance and its demonstration with a suitable newly developed or existing flying platform, focusing on three central objectives: (a) reducing manpower efforts in particular for large constellations, (b) reducing the number of false alerts, (c) reducing the time between maneuver decision and close approach [17]. Previously in 2014, the EU had established the Space Surveillance and Tracking Support Framework to develop European SST capability and form the SST Cooperation.

3. DISTRIBUTED LEDGERS TECHNOLOGY

Distributed Ledger Technology can be thought of as a technology of building distributed databases without a central physical component and point of failure. It is a system that automatically generates proofs of data integrity and proofs that any user of the database will see exactly the same data records at any given time point. The core principle behind *Distributed Ledger Technology* (DLT) is the Ledger that is being distributed/shared between all the users (hence, the name *Distributed Ledger Technology*) and upon which the consensus is reached. This means that, by following some well established consensus protocol, everyone eventually agrees what the ledger contains and that there is no alternative version of the ledger. The ledger contains a timeline of so-called *transactions*, data recording/updating/deleting events. Hereby, if every user follows this timeline and applies transactions in the same agreed upon manner, then everyone should have reconstructed the exact same replica of their local database at any given time point.

Blockchain is one of the most straightforward and common ways of implementing DLT. Hence, the terms DLT and blockchain are being used interchangeably in the literature. However, strictly speaking, blockchain is a type of DLT, where ledger is represented as a sequence of cryptographically linked blocks, and a block is simply a group of transactions that follow transactions from the preceding block. The very first real-life DLT implementation is well-known digital cash decentralized network Bitcoin [14]. Bitcoin (and other cryptocurrencies) is a permissionless network, meaning that network users use permissionless consensus protocol to reach global single view on the ledger. Here, the word *permissionless* means that any anonymous user can join, read and submit trans-

actions and leave the network any time. The drawback of permissionless consensus protocols is their relatively high complexity, non-finality (single-view guarantee on a data point is close to, but never 100%, it grows with time. That means that the data can be actually reverted, but usually with low probability, and with time this reversion is less probable) and in a number of cases unreasonable high physical resources and energy consumption cost [13]. On the other hand, anonymity and permission-free nature of DLT systems can be traded for instant finality, simplicity and low energy consumption in majority of business cases. Hereby, DLTs based on permissioned consensus are much faster and generally more secure than DLTs based on permissionless protocol, but they lack anonymity and free permission access for their users. Thus, depending on business case, one may choose between permissioned and permissionless DLT framework.

In this case, the focus was not on anonymous permissionless access to SDTaaS but rather on fast operations and data finality, hence DLT solution based on permissioned consensus was needed.

4. DLT FOR SPACE SURVEILLANCE AND TRACKING

The DLT key value proposal for SST domain is to enable secure data sharing across entities to have a clear consensus on any SST Catalog Data, without the system being controlled by any single entity and yet still acting as a single system in terms of being synchronised across all participants.

SST Catalog Data was chosen because "the heart of an effective space surveillance system lies in the comprehensiveness and accuracy of the space object catalog. The process to populate and maintain this catalog is not a trivial exercise and requires the manipulation of heterogeneous observation data to both perform orbit determination and correlation with objects already identified. The process should be sufficiently robust to allow the automatic incorporation of new objects as well as alerting for lost objects without a large overhead in terms of manpower and human interaction. [11].

The SST system aggregates data from multiple data sources. The SDTaaS system prototype is aimed to enable secure data aggregation from those sources. It does not include reconciliation between different datasets but will display all different orbital data unmodified, as it was submitted by different parties; the users will be able to see who submitted each dataset. The solution will confirm that no data has been tampered with after signing by the data providers and ensures that everyone gets the same data.

The system will remain available provided that sufficient amount of parties keep their servers available - as is typical with blockchain systems, even if a user is the only one running a working node (server), they can still read and query the data because they have a local copy of everything. However, to issue global updates, generally it is required that over two thirds of the nodes are operational and can send data to each-other. Such thresholds can be configured, however; different participants can also have different weights to their participation. In particular, even if a user is the only one running a working node (server), they can still read and query the data because they have a local copy of everything.

Already today (and likely more so in the future) many entities wish to aggregate orbital data from multiple sources by themselves (without relying fully on third parties), to feed orbital data into their operational infrastructure. However, every participant synchronising data by themselves without using DLT leaves serious risks untapped.

The most obvious risk of performing data aggregation without global (DLT) consensus on it lies in conflicting SST data being provided to different entities, either accidentally or on purpose by a malicious actor. Here DLT functions as a mechanism of synchronisation across all parties without relying on a single global party for its functioning.

Additionally, without signatures, catalog and satellite operators could falsely deny sending data or to claim sending various data, whereas signatures present verifiable evidence on data having been signed off (known as the non-repudiation property). For example, if a satellite operator has signed off on maneuvering plans, only to later deny ever agreeing to it, digital signatures can be used to prove to third parties the operator really saw the maneuvering plans and even explicitly agreed to them. Whilst DLT technology can ensure the integrity of data after being digitally signed, the author of the data can still provide false data. For now, we trust that the counterparties are motivated to provide correct data, especially if the data is digitally signed and/or published on the blockchain, thus making sure they cannot deny the act of sending or receiving any relevant data.

DLT can also verify other integrity properties besides verifying signatures on proposed action plans. For example, a satellite operator could attempt to sign a proposed maneuvering plan but would never share it. If anything goes wrong, it could later claim that they did send their plan to other parties and they could even show the signed message they supposedly have sent. Consequently, it will claim the other parties received the message but choose to pretend that they did not. Using DLT protocols, it can be proved the message was never received by the other parties (either the specific recipients or the DLT consensus as a whole), thus quickly disputing their claims of sending such messages.

DLT is also to be used for resilience and performance improvement although for the use case of conjunction predictions, high performance is not most critical since decisions are made well in advance. However, as an example, without using digital signatures on orbital data, parties cannot confidently forward data to each other in case the

original data source is temporarily offline as there is no means to verify the integrity of the data. Thus, it would take longer to receive and verify the data before it could be trusted.

Even though at this stage storing false data cannot be prevented, it can be ensured that the data stored in the blockchain complies with certain data standards. The SDTaaS prototype does this by enforcing the use of The Consultative Committee for Space Data Systems (CCSDS) recommended standards [4] in smart-contracts which validates the data compliance before committing the Catalog Data in the blockchain. We believe this to be a valid instrument to prevent discrepancies among participating entities on the data formats. The formats selected for the SDTaaS prototype are Two-Line Element (TLE), Orbit Data Message (ODM), Fragmentation Data Message (FDM), Conjunction Data Message (CDM) and Reentry Data Message (RDM). Although this list can easily be extended, the chosen formats prove that the solution can handle different data payload sizes as well as operation under varying data volumes.

5. THE SDTAAS PROTOTYPE

The SST Data Value Chain [9] consists of three main functions: data detection, data processing (cataloging) and information provision. The detection function consists of a network of sensors to survey and track space objects in all orbital regimes. The processing function aims to coordinate the data-sharing, processing and analysing of shared data. The information function oversees providing the SST services to end users. The SDTaaS system prototype serves to complement the data processing (cataloging) function of the value chain, enabling multiple SST Catalog Data providing entities to form up a decentralized network for sharing SST Catalog Data with each other, thus benefiting from the advantages provided by the blockchain technologies.

5.1. Hyperledger Fabric

SDTaaS prototype was developed using *Hyperledger Fabric* (HF) framework [8, 12]. HF allows to create solutions with a modular architecture with pluggable consensus and membership services. It provides support and a Software Development Kit for chaincode (aka. smart contracts in HF terminology) development in a range of industry-grade popular languages and environments (in this case Java was used). HF enjoys high security and efficiency due to its transaction endorsement mechanism: for any transaction being submitted to HF blockchain, it should first be endorsed/approved by the majority (or certain well-defined group of) organizations participating in the network, hence eliminating situations for possible short-term forks (split-view on the data by different network members), assuring data integrity (i.e., data being validated according to well-established rules) and instant finality (once added to the ledger, data point will never be modified/reverted) of all the entries landing to the ledger. Also, HF comes with out-of-the-box database solution (CouchDB) allowing to form complex SQL-like queries against the data in the ledger.

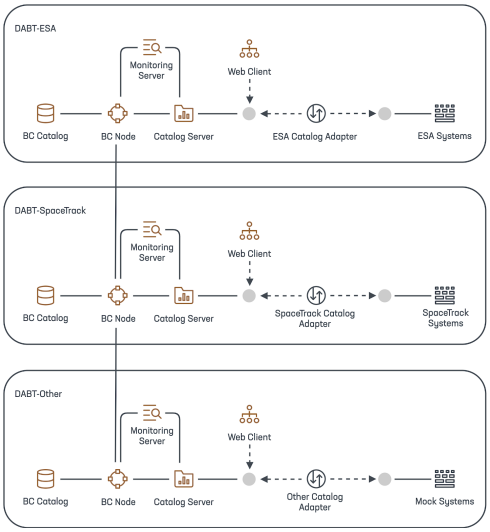

Figure 1. High-level architecture of SDTaaS systems for identified entities.

5.2. SDTaaS System Architecture

The SDTaaS system is represented by multiple application tiers inspired by [2]. The identified tiers for SDTaaS system components are business, web and client tiers. Business tier components are responsible for business logic that delivers the domain specific functionality. Web tier components are responsible for interacting with client and business tier components. Client tier components have the functionality to make requests to web tier components and process the responses received from them.

Figure 1 shows the overall architectural design of the SDTaaS system, which represents the deployments of each component per identified entity.

The Blockchain Node (BC Node) component contains the blockchain data peer with the main responsibilities of maintaining the connections between other BC Node peers located in other entities and also serving the possibility to run deployed smart contracts. The component is part of the business tier and deployed as multiple Docker [5] containers containing a running instance of Hyperledger Fabric server. The key functionalities of this component can be summarized as:

- Maintaining a closed blockchain network that only the participants with correct identities and permissions can join.

- Controlling access to the deployed smart contracts. These smart contracts are used to create, update, delete and read Catalog Data and Entity information, and update and read relevant metrics information regarding the operations performed per entity.
- Providing visualization of the transactions that form the blocks inserted in the blockchain.
- Replicating Catalog Data and synchronizing the blockchain transactions across all the participants.

The Catalog Server component is the main interaction point for all the blockchain related operations. Mainly responsible for serving the queries that are used to store, retrieve, and update the Catalog Data that are maintained within the blockchain (by the BC Node components). The component is part of the web tier and deployed as a Docker container containing an executable web server written in Java with the following key functionalities:

- Serving requests related to Catalog Data operations received from the users and Catalog Adapter components.
- Connecting to the assigned BC Node component and triggering deployed smart contracts.
- Providing metrics regarding its resource and operational usage.

The Monitoring Server component is responsible for monitoring the Catalog Server and the BC Node to give information regarding the operational status of these server components. The component is part of the business and web tier and deployed as multiple Docker containers containing executable server and client applications. Key functionalities of the component are to:

- Provide detailed metrics related to the resource usage of Catalog Server and BC Node components.
- Provide Java Virtual Machine (JVM) related metrics of Catalog Server component.
- Provide data I/O rate metrics of the Catalog Server component.

The Web Client component is for visually displaying the contents of the BC Catalog and validating supported external Catalog Data to see if it exists in the blockchain. The component is part of the client tier and deployed as a standalone client application in a Docker container written in Javascript, HTML and CSS. Key functionalities of the component are to:

- Create, update, delete and query Catalog Data in the blockchain.
- Validate the existence of a Catalog Data by checking if the computed hash of the data has been inserted in the blockchain.
- Retrieve human readable metrics regarding the Catalog Data related operations performed by individual entities.

Besides using the Web Client for advanced querying and data existence checks, data audits can also be performed using a separate tool called Hyperledger Explorer [12] deployed optionally as part of the BC Node component. This tool can be used to display raw data for each committed transaction that in turn form up the created blocks.

Integrating the SDTaaS system prototype with the ESA's SST Core Software System and Spacetrack's web services in real time was not a feasible option. Therefore, a decision was taken to develop mock versions of these services that mimic the generation of the supported Catalog Data formats (see Fig. 1). These components are part of the business tier which are deployed as multiple Docker containers that contain a Web Application Resource (WAR) distribution deployed in a tomcat server, and a MySQL database, where information about generated records will be stored. The functionalities of these mock services can be listed as:

- ESA System mock mimics SST Core Software HTTP SOAP service definitions for fetching ODM, FDM, CDM and RDM Catalog Data formats.
- Spacetrack System mock provides an HTTP REST interface for generating OMM and TLE Catalog Data type.

The Catalog Adapter is used to integrate the SST Catalog Provider services with the SDTaaS system without modifying any of the core software. It can be used to develop customized services based on the operations supported by the SDTaaS system depending on the use-case. Two distinct Catalog Adapters were developed as part of the prototype software to mimic a possible workflow of managing SST Catalog Data in the blockchain for ESA and Spacetrack mocked services. This component is part of the web tier and deployed as a Docker container containing an executable command line based client application written in Java. The key functionalities of the component are to:

- Make periodic HTTP based requests to the services provided by the SST Catalog Provider Systems (ESA System mocks and Spacetrack System mocks).
- Route the responses received from the SST Catalog Provider Systems to its respective Catalog Server with the expected data structure.

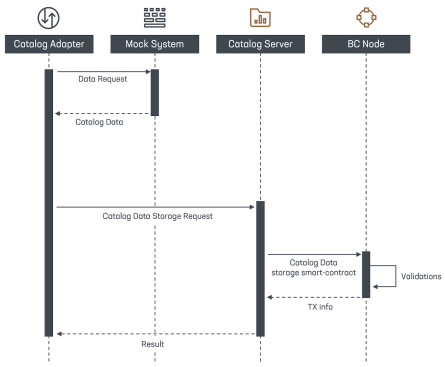

Figure 2. High-level UML sequence diagram of flow of operations for automated Catalog Data storage

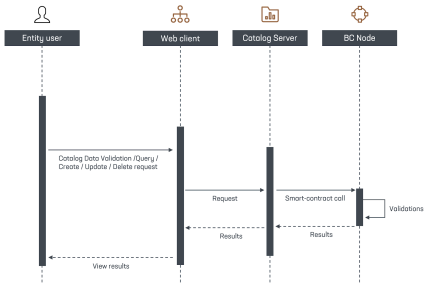

Figure 3. High-level UML sequence diagram of flow of operations of an authenticated user

5.3. Concept of Operations

The SDTaaS system prototype software aims to handle the Catalog Data storage procedure in an automated workflow as shown in Fig. 2. Data flow starts from the Catalog Adapter, where the adapter periodically asks for new data from the catalog providing system. As soon as it receives the Catalog Data from the system, the adapter wraps the Catalog Data in a storage request and routes it to its assigned Catalog Server. After validating the request parameters, Catalog Server routes the request to the BC Node by calling the smart-contract responsible for storing the data in the blockchain. Using the smart-contract, BC Node validates the Catalog Data portion of the received request against its respective data schema definition generated from CCSDS Recommended Standards [4] and SpaceTrack data model definitions [6, 7]. It then stores the data in the BC Catalog replicated across all the participating entities. Finally, the transaction information is returned to the Catalog Server and the storage result is routed back to the Catalog Adapter.

Apart from the automated workflow, an authenticated user can perform any supported SST Catalog Data related operation by interacting with the Web Client's User Interface (UI). As shown in Fig. 3, Web Client is responsible for making an API request to the Catalog Server. After making necessary request validations, Catalog Server makes the relevant smart contract call deployed on the BC Node based on the request. The smart contract execution results are sent back to the Catalog Server and finally are routed back to the Web Client which displays the responses on the UI.

In addition to the functional requirements, governance features have also been considered. Software updates and onboarding of new participating entities have been handled using a set of defined policies in the HF framework which allows all members to have control over maintaining the system environment collectively with the decisions accepted by the majority of participants.

6. DISCUSSION

For the prototype software, the raw Catalog Data was stored on-chain using Hyperledger Fabrics internal distributed document store called CouchDB. However, this approach proved to be not the most efficient in terms of overall system performance and resource utilization. A more suitable approach would be to store the calculated hash of the record, incl information about the transaction which would already guarantee that the submitted data is endorsed by the majority of the participants, in the blockchain. The raw Catalog Data would be stored in a separate database component deployed per each entity. The separate database component can either be a relational database or a distributed document store. For this, however, further analysis is needed.

Moreover, if each entity stores only their own data in the separate database component and shares it on demand with others (i.e., organize a federated database with each datapoint validated by the majority of the entities, but stored only on one or few entitys nodes), data storage footprint for individual members would be considerably reduced.

An additional option would be to deploy the current approach by storing all data in the blockchain but with another component created to pull data from the blockchain, either periodically or with a notification mechanism, to then save it in a relational database or a document store deployed separately. This separate database and component would enable to query data much faster. However, the issues with disk space utilisation remain not being mitigated.

More options for scaling SDTaaS platform remain to be investigated and other blockchain solutions besides HF should be considered as well. In Tables 1, 2, we present short comparative analysis of HF vs other blockchain solutions.

Table 1. Hyperledger Fabric vs alternative blockchain solutions

	Hyperledger Fabric	**Alternative solutions**
Data Integrity Validation Strategy	**Execute-Order-Validate strategy**: a transaction is being first endorsed (pre-consenus validation) by majority (or a required set of peers defined in the endorsement policy), then transactions are being ordered (consensus on ordering) and inserted into the ledger in strict order and broadcasted to all the peers, then each peer validates the ordered transactions (post-consensus validation) against the endorsement policy.	**Order-Execute strategy**: a transaction is ordered, then executed. Transaction is validated by the block creator (miner) before being included into the new block (transaction being ordered), then every member verifies transactions from the new block independently from each other (transaction being executed). A member considers new block valid if all transactions in it are valid
Advantages	Higher security: if we assume that majority of consortia members follow honestly the data integrity protocol (i.e., verify the new data by well defined rules and honestly report the result), then a situation even with short term split-view on the data (different members see alternative versions of the data) is not possible. Every transaction carrying signatures by majority endorsers (members who verified the transaction) is guaranteed to be valid and have no alternative versions. No need to validate transactions that have been already included into the blockchain	Straightforward transaction submission process: a transaction creator simply broadcasts the new transaction into the network, then the miner peaks up the transaction, verifies it and includes into the new block Relatively low traffic: new transactions are disseminated among all the members via gossip protocol Lower storage footprint (no need to keep endorsements along with every data point) Simple architecture
Disadvantages	Complex transaction submission process. First of all, a transaction creator has to communicate with the majority of consortia members, send to them the transaction proposal and to collect their endorsements. Then, the transaction with the endorserers signatures can be submitted into the blockchain for the inclusion High traffic during transaction submission. Total traffic volume grows quadratically with the respect to the number of consortia members Complex architecture	Lower security: cannot rely on honest majority, all transactions must be verified by every member in order to rule out any faulty transactions included by dishonest/malfunctioning miners. Short-term forks are possible, since there may exist two valid alternative transactions in alternative blocks created by two different (or even the same) miners
Mitigate the disadvantage	Redesign endorsement policy: Rather than requiring simply to endorse transactions by the majority, define a smaller subset of members who can endorse (vote for the valid transaction) and their voting weight	Wait sufficient amount of time to catch all possible alternative blocks disseminated through the network via the gossip protocol, and/or Rely on the permissioned consensus protocol that can establish instantly finality of the block (may still involve majority voting) In case next miner is chosen deterministically, assume honest miner of the block (assume that the miner does not create alternative blocks)
Governance (membership management)	In-system defined policy	Policy implemented within the chaincode
Voting	In-system defined policy	Policy implemented within the chaincode
Weighting of controlling participants	In-system defined policy	Policy implemented within the chaincode
Trust in data validity	Reduced to trust to the consortia majority	Achieved via independent verification of every datapoint by every entity

Table 2. Hyperledger Fabric vs alternative blockchain solutions (cont.)

Off-chain data storage (store on-chain hash of the data record, but keep the data record itself off-chain in a database)	Straightforward implementation of trust, since the data is verified by the majority prior its submission to the blockchain. Benefits: save individual storage by deciding what raw data to keep at which entity	More research needed regarding the trust to data records, since all the entities must see and verify all the data points independently from each other anyway. Benefits: even though all data points must be seen by all the entities, still one can prune raw data from local storage as soon as it was verified. An advantage over Fabric solution: less storage requirements, since no endorsements associated with every data entry to be stored within the blockchain.
Network communication requirements	High: Quadratic complexity with respect to the consortia size	Lower: could be linear
Storage footprint	Higher: need to store endorsements by consortia majority for every data point	Lower: store just raw data with authors signature and usual block data structures
Security	Much higher: instant finality of data in the blockchain	Lower: possible short-term split-view on new data
Efficiency	Can be configured to be highly efficient with smart policy management and distribution of tasks between fabric components.	Harder to scale-up: normally every entity performs the same tasks. Not possible to split tasks between different entities.
Network administration	Much more complex due to complex Fabric architecture	Can be configured for almost zero administration

7. CONCLUSION

The high-level user needs identified for the SDTaaS prototype system included:

- Establishing integrity, availability, authentication and clear consensus on SST data provided by multiple data sources.
- Using SST data for automating planning and maneuvering decisions based on shared rules / operating concepts across multiple operators to ensure a reliable detection and collision avoidance.

The role of DLTs in addressing those was identified as following:

- To pave the way for decentralised, secure, and trusted SST data sharing platforms, resulting in a system not controlled by any single entity, yet acting as a single system, synchronised across all participants and application areas.
- To establish long term trust and secure exchange mechanisms between space operators by deploying fully traceable information on future trajectories of space objects, on execution of maneuver plans to detect and prevent collisions and by enforcing rules (e.g. via smart contracts) for acknowledgement of collision risks and subsequent negotiated maneuver execution.

It is clear from above that in order to make decisions based on the data, the confidence on the completeness, accuracy, as well as integrity and availability of the data shared must first be established. The 2nd point of the high-level user needs above therefore depends on the availability and success of the 1st point. The SDTaaS prototype in return, aims in proving that the 1st point can work as expected and upon successful completion the goal is to move forward with addressing the 2nd point in the future.

Once long term trust and secure exchange mechanisms have been established, SDTaaS prototype can feed into various business logic applications that can be built on top, such as smart contracts for Automated collision avoidance manoeuvring.

Today, active collision avoidance has become a routine task in space operations, relying on validated, accurate and timely SST data. With catalogs set to expand in future years owing to improved sensor precision and data processing, the task of committing to classical human led assessments, however, will prove to be unmanageable due to increasing workload demands [10].

Here the long-term target is a system which enables the automated and immutable exchange between operators of the results of conjunction analyses and the execution of critical multi-party procedures to be enforced. Based on the common dataset, DLT could ensure that future agreements and plans for manoeuvring between operators are exchanged and operated on defined, enforced rules via

smart contracts. Smart contracts on DLT can create a provable audit trail of, for example, who agreed to perform which manoeuvre, and more generally, who sent which messages to whom. If the expected exchanges are enforced by a smart contract, the sender can also be immediately notified if the receiver does not acknowledge the receipt, and the smart contract can decide whether both parties have correctly responded. This can be a key mitigation to issues with manual exchanges and human errors, such as the 2019 ESA collision avoidance manouevre with a SpaceX satellite in the Starlink constellation.

In the future, the system could also accommodate sharing of more private data sets. Anonymous participants may also be supported, for handling collision cases with classified objects. In this case, a satellite operator could be conversing with an unknown national entity but other members of the conversation could still be verifiable by everyone if needed.

Also, both large international smart contracts between many parties as well as smaller ones between only two entities can be deployed. The smart contracts could be collectively governed by international governance bodies, depending on the scope of each contract.

One caveat may be that data providers may nonetheless manipulate their own respective data, either intentionally or not. Hence, in case of an incident, the chain of communications can be analysed along with any decisions agreed by smart contract and responsible entities can be traced. In addition, if ephemeris data are uploaded, an immutable record of compliance (or not) with space debris regulations/policies can also be ensured.

REFERENCES

1. Trusat white paper. Consensys Space. URL: https://trusat-assets.s3.amazonaws.com/TruSatWhitePaper_v3.0.pdf.
2. Tiered applications (your first cup: An introduction to the java ee platform), 2010. URL: https://docs.oracle.com/cd/E19226-01/820-7759/6nj0dg0f8/index.html.
3. Espi report 71-towardsa european approach to space traffic management-full report. European Space Policy Institute (ESPI), January 2020. URL: https://www.espi.or.at/.
4. Blue books: Recommended standards, 2021. URL: https://public.ccsds.org/Publications/BlueBooks.aspx.
5. Docker overview, 2021. URL: https://docs.docker.com/get-started/overview/.
6. Space-track data model definition of the general perturbations (gp) class, 2021. URL: https://www.space-track.org/basicspacedata/modeldef/class/gp/format/html.
7. Space-track data model definition of the two-line element (tle) class, 2021. URL: https://www.space-track.org/basicspacedata/modeldef/class/tle/format/html.
8. Elli Androulaki, Artem Barger, Vita Bortnikov, Christian Cachin, Konstantinos Christidis, Angelo Caro, David Enyeart, Christopher Ferris, Gennady Laventman, Yacov Manevich, Srinivasan Muralidharan, Chet Murthy, Binh Nguyen, Manish Sethi, Gari Singh, Keith Smith, Alessandro Sorniotti, Chrysoula Stathakopoulou, Marko Vukolic, and Jason Yellick. Hyperledger fabric: A distributed operating system for permissioned blockchains. 01 2018.
9. Duke Charlotte, Flytkjaer Rasmus, Esteve Romain, and Oswald Nick. Commercial space surveillance and tracking market study. London Economics, July 2020. A report for the UK Space Agency. URL: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/917911/LE-UKSA_Commercial_Space_Surveillance_Tracking_FINAL_FOR_PUBLICATION.pdf.
10. ESA. Esa spacecraft dodges large constellation, 2019. URL: https://www.esa.int/Safety_Security/ESA_spacecraft_dodges_large_constellation.
11. Emmet Fletcher. From debris to database: the development of an efficient data processing chain for space situational awareness services. In *SpaceOps 2012 Conference*. American Institute of Aeronautics and Astronautics, jun 2012. doi:10.2514/6.2012-1287629.
12. Hyperledger White Paper Working Group. An introduction to hyperledger. Hyperledger, Technologies for Business, August 2018. URL: https://www.hyperledger.org/wp-content/uploads/2018/08/HL_Whitepaper_IntroductiontoHyperledger.pdf.
13. Jingming Li, Nianping Li, Jinqing Peng, Haijiao Cui, and Zhibin Wu. Energy consumption of cryptocurrency mining: A study of electricity consumption in mining cryptocurrencies. *Energy*, 168:160–168, feb 2019. doi:10.1016/j.energy.2018.11.046.
14. Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2009. URL: http://www.bitcoin.org/bitcoin.pdf.
15. Joseph N. Pelton. Us space policy directive-3: National space traffic management policy, 2020. doi:10.1007/978-3-030-20707-6_100-1.
16. Harvey Reed, Nathaniel Dailey, Robert Carden, and Dave Bryson. Blockchain enabled space traffic awareness (besta). 10 2019.
17. Benjamin Bastida Virgili, Tim Flohrer, Holger Krag, Klaus Merz, and Stijn Lemmens. Cream - esa's proposal for collision risk estimation and automated mitigation. *First International Orbital Debris Conference*, 2019. URL: https://www.hou.usra.

edu/meetings/orbitaldebris2019/orbital2019paper/pdf/6031.pdf.

18. Waqar Zaidi, Weston Faber, Thomas Kelecy, Naeem Altaf, and Sowmya Janakiraman. Enabling worldwide and transparent space traffic management through decentralized and trustworthy space domain awareness. *18th IAA SYMPOSIUM ON SPACE DEBRIS*, A6, 2020. URL: http://iafastro.directory/iac/paper/id/60949/abstract-pdf/IAC-20,A6,10-B6.5,7,x60949.brief.pdf?2020-07-06.15:32:06.