Design Platform for Multi-Agent Communication System for Resilient CubeSat Swarms

William Edmonson [(1)], Miguel Nunes [(2)]

[(1)] Sadaina LLC, Norfolk, VA 23505, +1 919.757.6205, wwedmonson@sadaina.com
[(2)]Hawaii Space Flight Lab, Honolulu HI 96882, phone +18089560441, manunes@hsfl.hawaii.edu

ABSTRACT

Satellite swarms are required in order to improve the temporal, spatial, and spectral resolutions of Earth Observation missions. To ensure coordinated operations of CubeSats deployed as a SWARM for achieving mission operations requires a resilient inter-satellite communications sub-system, we propose using a LED-based visible light communication system to improve swarm communications. This system also meets the SWaP constraints of pico-/nano-class of small satellites. The focus of this paper is to develop a modeling and simulation environment that ensures specifications are met throughout the design and build phase of the inter-satellite communication system. The proposed development environment consists of integrating the Comprehensive Open-architecture Solution for Mission Operations Systems (COSMOS) simulation testbed with the Arcadia/Capella Model-based Systems Engineering (MBSE). This development environment will allow for testing of VLC systems in a swarm inter-satellite communication example given a swarm mission consisting of 3 child satellites communicating with a mother satellite. An MBSE model of this example will also be shown.

1 INTRODUCTION

Many small satellite university missions have seen long development times and/or during on-orbit operations many of those missions have failed, e.g., between the years 2002 and 2016 there have 139 failed missions out of a total of 270, these statistics are for CubeSat's [1]. Much of this failure could have been mitigated if the design teams have followed the system engineering process and/or simulated various aspects of systems of missions at multiple phases of the design. It has also been shown that if the design team implemented a model-based system engineering methodology, could reduce the overall design process, improve communication amongst the design team, and provide a direct connection between models and simulation. By implementing systems engineering and simulation into the university curriculum will give the students much-needed expertise and experience to better design, build, and implement more complex systems. These skills are in demand by industry - to be able to think and process systematically.

> *Systems Thinking is a discipline for seeing wholes. It is a framework for seeing interrelationships rather than things; for seeing patterns of change rather than static 'snapshots'. ... Today systems thinking is needed more than ever because we are becoming overwhelmed by complexity* [2].

Present satellite systems engineering process and methodologies do not translate very well to pico-/nano-class of satellites (PNSats) because of the cost and time incurred due to the multiple design reviews, the larger cost associated with the asset, additional time for design, and the overall increase in complexity of the design due to the drive to integrate multiple and sometimes disparate instruments on the same platform. The latter requires many more engineers along with an increase in validation and verification assurance (V&V) checks. Because of this higher cost associated with the design and build of the large space assets, a risk-adverse design philosophy is practiced. The above set of reasons provides motivation for mitigating complexity through the use of swarms.

COSMOS-Swarm is being developed via a NASA STTR project between Interstel Technologies and the University of Hawaii, the COSMOS-Swarm Phase II was recently initiated in the Spring of 2022, to enable scalable mission control for multiple diverse assets simultaneously such as terrestrial or planetary Earth, lunar, or Mars missions with diverse NASA assets including aerial, ground, subterranean, and underwater agents. Example missions include, but are not limited to, monitoring of large-scale dynamic events (e.g. wildfire outbreak) with convoy agents and following sensing agents or the formation and maintenance of leader-follower satellite constellation for volumetric remote sensing. The swarm formation consists of three or more satellites orbiting the same or adjacent orbits. These satellites form a virtual topology structure according to their mission requirements. The swarm topology can provide an image of the ground target from different angles and positions at the same time or across different times increasing the swath width, time resolution and other important metrics. In COSMOS Swarm the Mothership determines and controls the positions of the ChildSats. NASA's Technology Roadmaps [14] which denes the pathways for this sector of technology research and development for the next 20 years (2015-2035), and different decadal surveys [16, 24, 25, 26, 27] determined that swarm missions that can collect simultaneous measurements from a 3D volume of space are very important for a variety of missions for Earth Science and Space Physics. Also strong commercial opportunities exist for swarming missions as well to support Earth observation commercial activities to atmospheric monitoring, etc.

This paper will layout the development and simulation environment for the design of an inter-satellite communication subsystem. This environment integrates the Comprehensive Open-architecture Solution for Mission Operations Systems (COSMOS) simulation testbed with the modelling process and tools of Arcadia/Capella. Though following this design path will require significant effort in the initial stages of the design, the result will be a reduction in the time and effort during the later validation and verification phase. Most importantly it will reduce the number of redesigns during these later phases and thereby resulting in keeping within the cost constraints.

2 BACKGROUND

2.1 Visible Light Communication for Inter-Satellite Links

Small satellites, deployed as a sensor network in space, have an advantage over conventional satellites in space exploration because of their potential to perform coordinated observations, high-resolution measurements, and identification of Earth's asset that is inclusive of its space environment. Low-latency communications between these satellites result in improved availability for observation, telecommunications and reconnaissance applications [5]. Presently, the dominant research and development for implementing inter-satellite communication links (ISLs) consists of using either radio frequency (RF) or highly directed lasers. The latter will require a highly accurate pointing satellite control system, while the former is not suitable for systems with sensitive electronics onboard or in applications where high data rates are required due to the limited available spectrum. It is also mechanically challenging to deploy large parabolic antennas on small satellites equipped with RF radios in order to support high data rates. The required pointing accuracy needed for laser communication presents a challenge to the CubeSat form factor represented by the pico-/nano class of satellites due to the stringent SMaP restrictions imposed by the platform. Lasers produce a narrow and focused beam of light that could fall out of the field-of-view (FOV) of a small satellite receiver due to slight movements. Furthermore, for formation flying systems in LEO, the ISLs are much shorter than links between satellites in geostationary orbit; thus, the use of lasers and the highly accurate pointing they provide can be considered superfluous [8,9]. To minimise the SMaP constraints imposed by the platform, along with the need for reduced pointing accuracy and to achieve high data transmission rates, we propose a visible light communication (VLC) subsystem for pico-/nano class of satellites for ISC. These multiple small satellite missions will benefit from VLC's ability to transmit higher data rates with smaller, light-weight nodes, while avoiding the usual interference problems associated with RF, as well as the apparent radio spectrum scarcity below the 6 GHz band [10]. Furthermore, the electronics required for achieving precision pointing accuracy for laser communication systems will be avoided. With approximately 300 THz of free bandwidth available for VLC, high capacity data transmission rates could be provided over short distances using arrays of LEDs. See Fig. 1 for VLC system block diagram.

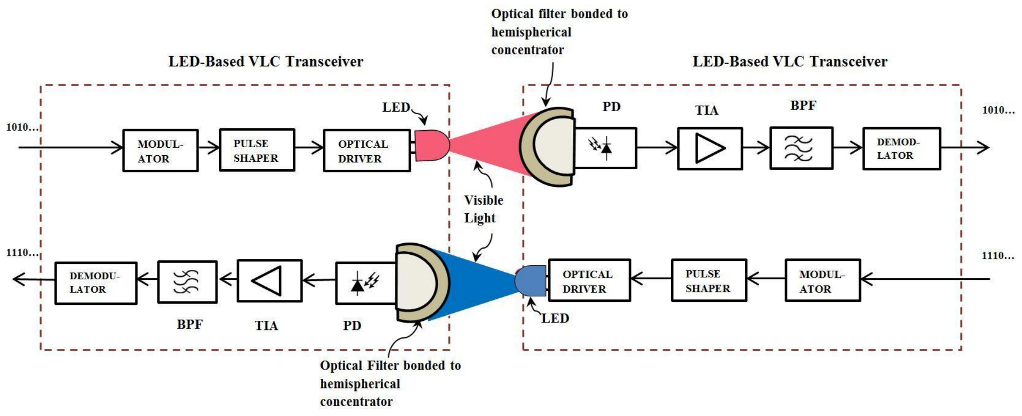


Fig. 1. Block Diagram of VLC System

2.2 VLC Link Budget

We can model the line-of-sight (LOS) link between any two adjacent satellites in a trailing formation or within a cluster according to the generic LOS VLC scenario shown in Fig. 2. The distance between the LED emitter and detector is denoted by d, while the angle of incidence with respect to the receiver axis is ψ, and the angle of irradiance with respect to the transmitter perpendicular axis is φ. The viewing angle φ indicates how focused the beam is when emitted from the LED.

In line-of-sight (LOS) optical links, the relationship between the received optical power P_r and the transmitted optical power P_t can be represented by Eq. 1\cite{komine, amanor}. The quantity $H(0)$ represents the channel DC gain and it is the single most important quantity for characterizing LOS optical links.

$$Pr \quad = H(0)Pt \tag{1}$$

As shown in [9] and illustrated in Fig. 2, the channel gain in LOS optical links can be estimated fairly accurately by considering only the LOS propagation path.

$$H(0) \ = \ \frac{\beta}{2\pi d^{\ 2}} Ang(\psi, \varphi) \tag{2}$$

where β represents a gain term that includes the characteristics of the receiver photo-diode, $Ang(\cdot,\cdot)$ is a function of the LOS angles and transmission and concentrator gain. The link budget will be integrated into the COSMOS testbed to provide how the VLC transmission will behave within its operational environment. The noise sources associated with operations in space can be attributed to the Sun and to the characteristics of the receiver photodiode (PD), i.e., the shot noise and thermal noise, respectively. The electrical SNR at the receiver for measuring quality of communication is

$$SNR = \frac{S}{N} = \frac{(\gamma Pr)^2}{\sigma^2_{shot} + \sigma^2_{thermal}} \tag{3}$$

where γ represents PD responsivity.

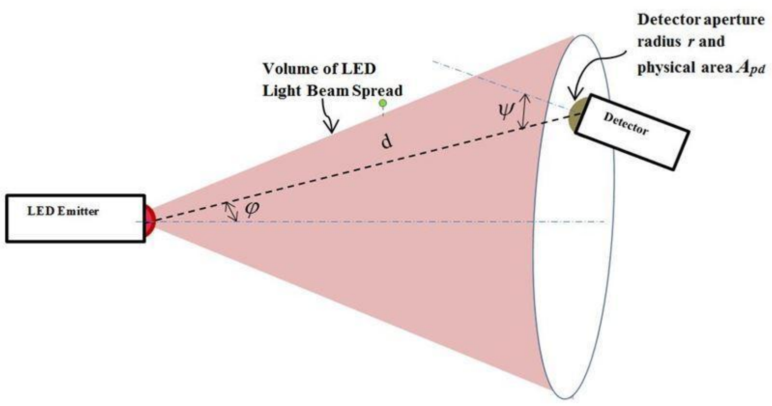

Detector aperture radius r and physical area A_{pd}
Volume of LED Light Beam Spread
ψ
d
Detector
LED Emitter
φ

Fig. 2. LOS VLC Link Model: Adapted from \cite {cui}

2.3 Model-based Systems Engineering

As space systems become more complex and more integrated, the design methodology must be able to compensate without extending the development life-cycle, provide better communication amongst the engineering disciplines, and increase fidelity of the complete systems design capture. The latter includes capturing the behaviour and performance of the system, its components, and their interactions while providing a direct connection at all levels of abstraction. Historically the design methodology has been somewhat ad-hoc and is based on a document-centric approach. A document-centric methodology conveys and stores information through documents, e.g., MS Office whereby each engineering or subsystem discipline conveys the associated performance and hardware behaviour characteristics using that discipline's language. Without a common language and/or semantics can lead to misinterpretation of the system specification at the subsystem interfaces, between the different levels of abstraction, and loss of historical data due to lack of appropriate data management. These issues can be mitigated through the use of a model driven design and the use of common design language and data management. Examples of such model-based systems engineering (MBSE) languages are the systems modelling language (SysML) [6] or Eclipse Capella and its associated design methodology Arcadia [12]. In the context of MBSE, a model represents an abstracted concept of a physical realisation with only relevant information, the components of these abstractions and the rules for the interconnections.

We have chosen the graphical modelling language of Capella [12], see Fig. 3, to be used as the basis of our design process development for small satellites and will be used for the design, analysis, simulation, and verification and validation of systems. It strongly supports MBSE practice. Another important aspect of Capella is that it facilitates communication by being a common language across organisational disciplines, thereby reducing ambiguity caused by a lack of a common formal design semantics.

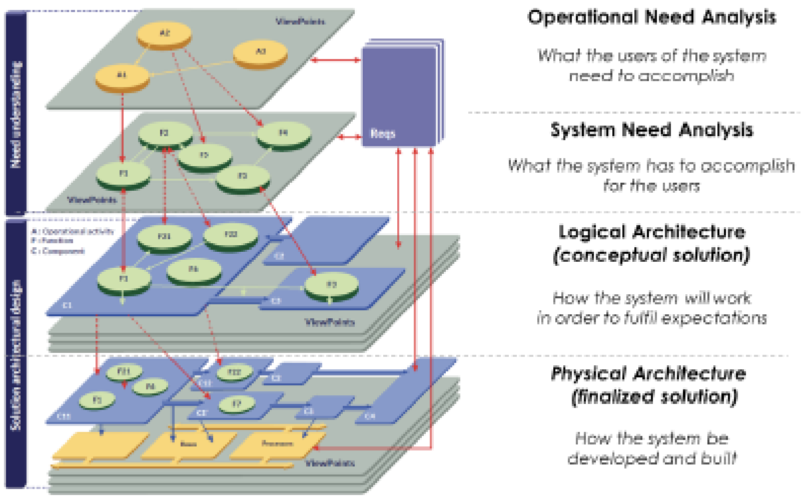

Need understanding
Solution architectural design
ViewPoints
Reqs
Operational Need Analysis
What the users of the system need to accomplish
System Need Analysis
What the system has to accomplish for the users
Logical Architecture (conceptual solution)
How the system will work in order to fulfil expectations
Physical Architecture (finalized solution)
How the system be developed and built

Fig. 3: The Arcadia Engineering Methodology

Model-based Systems Engineering has been used in the development of the VLC system for ISC as a research program. The modeling tool SysML was used [3].

2.4 COSMOS Overview

The Comprehensive Open-architecture Solution for Mission Operations Systems (COSMOS)is a software framework developed at the Hawaii Space Flight Laboratory for operating distributed robotic systems, with a particular focus on space systems such as CubeSats. It is also designed to operate satellite swarms and constellations. COSMOS is a software ecosystem with various applications that tie together the embedded flight software with a software agent-based framework, ground station management software, and the mission operations center user interfaces. It is intended to enable and facilitate SmallSat mission operations at universities and research organizations with limited budgets and short schedules. COSMOS is developed under the new paradigm of the internet-of-things where any asset can be connected to the system in a plug and play approach making it very generic for the inclusion and removal of assets. When an asset is connected and is COSMOS compatible, all the system can be informed of its presence and receive telemetry that can be processed to take actions. Figure 4 shows the range of exploration vehicles that COSMOS can be deployed on, the common interface are the software agents that coordinate the telemetry and control for the various nodes.

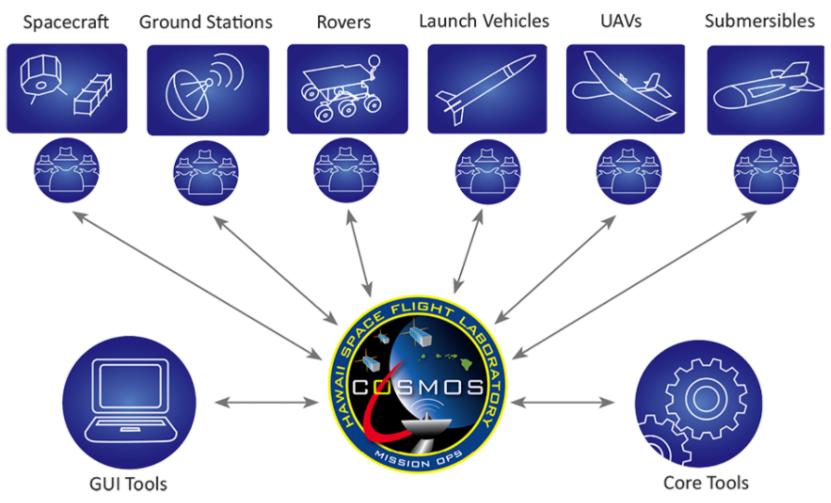


Fig. 4: COSMOS can be deployed in multiple remote mission applications with software agents.

The COSMOS software framework implements the various aspects of mission software; flight, ground, and mission operations. The key operational concepts of this Framework as it relates to the mission are:

- Web Standards: User interfaces are implemented over a standard framework on web technologies. All information data in COSMOS is represented in JavaScript Object Notation (JSON), both on disk and over the network. The Command and Control are implemented as native OS programs (clients and servers). The data is transferred as either files or network packets. The network communication is performed via IPV4 UDP packets whenever possible.
- Generalization of hardware devices: The common properties of hardware devices (voltage, current, temperature, etc.) are generalized and wrapped with the software agents' state of health messages. The agents run as native OS servers on the flight computer or representative equivalent computers.
- Modularization: Satellite subsystems and/or devices are organized and controlled via dedicated agents. Specific agent functions are available as 'agent requests' and all the commands are accessible from the native OS command line. This provides a modular interface that can be combined in various forms depending on the mission's needs.
- Reuse of software components: software agents are used in every aspect of flight and ground operations. The software agents are constructed so as to operate in a common way at the system level while behaving similarly at the device level.

When deploying COSMOS on a node there is a common set of software agents to execute:

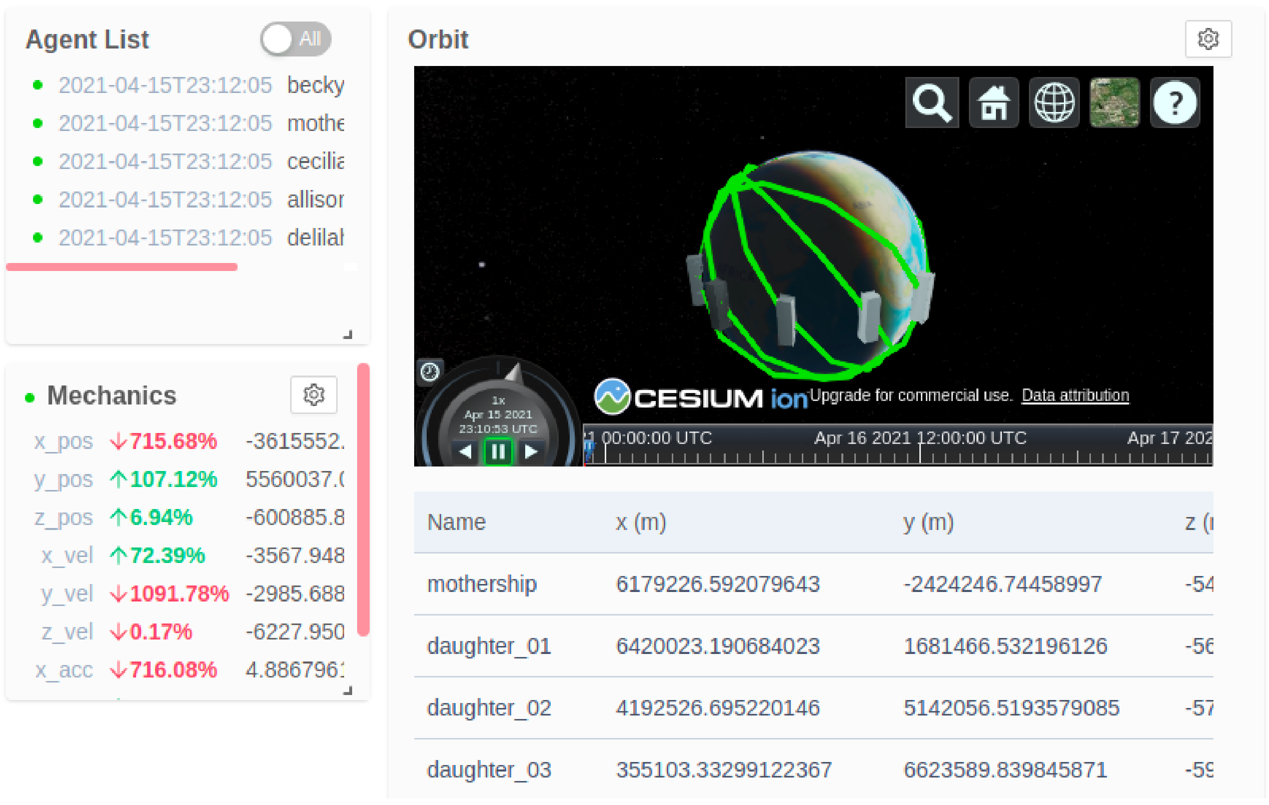
- Executive Agent (agent_exec) gathers State of Health (SOH) information from all other Agents, and manages system time the driven queue of uploaded and native OS commands. Also manages the logs for the system.
- File Transfer Agent (agent_file) manages the upload and download of files when there is a ground station contact. The underlying protocol uses UDP. This process is modeled after file transfer protocols such as the Saratoga File System and CCSDS File Delivery Protocol to be robust over intermittent connections. Files are selectively transferred to and from a set of standard Agent specific directories based on size and priority.
- Network Tunnel Agent (agent_tunnel) creates an IP network interface tied either directly to a serial port, or indirectly to a local network socket. Specific hardware Agents can then attach to this Agent to send and receive packet data across various radio interfaces.

Flight Agents:

- Electrical Power System Agent (agent_eps) implements the driver to the EPS system and controls power switches and reports power telemetry to other agents.
- Attitude Determination and Control System Agent (agent_adcs) implements the driver to the ADCS to command the ADCS unit and report attitude telemetry to other agents.
- Radio Agent (agent_radio) manages the transmission and reception of radio packets through the radios. This is tied to agent_tunnel to simulate an IP connection.
- VLC Agent (agent_vlc) manages the transmission and reception of data packets through the VLC system.

There are more categories of software agents for the ground station, mission operations center, and the operations testbed.

COSMOS Web is a NodeJS based web framework developed as a mission operations tool for the flight and ground station agents. The web interface utilizes the COSMOS agent's generalized functionality to display real-time telemetry data and launch agent requests. COSMOS Web is capable of live agent telemetry visualization, as shown in Figure 5, as well as historical data queries from the Database. The user interface is developed with modularity to be compatible with any COSMOS agent. It is built up from customizable widgets, each with a single function. The modularity and customizability of the interface allows COSMOS Web to be tailored to a particular mission without the need for redesign of the software.



Agent List All

●	2021-04-15T23:12:05	becky
●	2021-04-15T23:12:05	mothe
●	2021-04-15T23:12:05	cecilia
●	2021-04-15T23:12:05	allisor
●	2021-04-15T23:12:05	delilal

● Mechanics

x_pos	↓715.68%	-3615552.
y_pos	↑107.12%	5560037.(
z_pos	↑6.94%	-600885.8
x_vel	↑72.39%	-3567.948
y_vel	↓1091.78%	-2985.688
z_vel	↓0.17%	-6227.950
x_acc	↓716.08%	4.886796

Name	x (m)	y (m)	z (i
mothership	6179226.592079643	-2424246.74458997	-54
daughter_01	6420023.190684023	1681466.532196126	-56
daughter_02	4192526.695220146	5142056.5193579085	-57
daughter_03	355103.33299122367	6623589.839845871	-59

Fig. 5: COSMOS Web view of a satellite orbit simulation

COSMOS deployment. COSMOS can be installed in multiple target platforms, a regular x86 computer such as Windows 7 or 10, on Unix-based architectures such as Linux Ubuntu 16+, Fedora and also macOS, but more importantly, it can be deployed in ARM processors such as the Raspberry PI Zero and BeagleBone. COSMOS has been tested in flight computers such as the ISISpace OBC (400MHz 32-bit ARM9 processor, 64MB SDRAM), and the HSFL OBC (TI DaVinci DM3730 32-bit Arm Cortex-A8 800-1000 MHz). The baseline COSMOS core is designed to operate in a multi-threaded system and operating system kernel with a networking transceiver hardware (or networking software layer) using sockets on UDP. For single-threaded micro-controllers, it is possible to deploy micro-cosmos (a minimal version of cosmos) that uses a subsystem of the software agent functions.

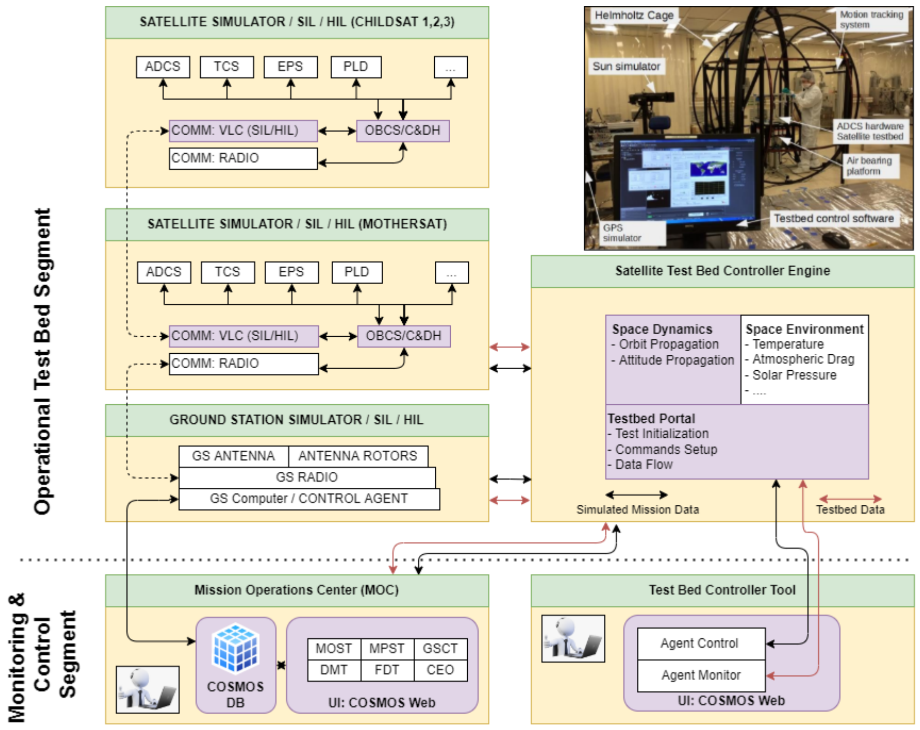


Fig. 6: COSMOS-VLC Operations and Simulation Testbed Architecture. Purple represents the VLC testbed elements that will be implemented.

The COSMOS-VLC test platform integrates the subsystems required for a representative mission simulation with the VLC channels and the satellite subsystems (ADCS, TCS, EPS, Telecom, etc.). These can be either fully implemented with the flight hardware or engineering models or even software simulated models, for use before the engineering models are produced. The COSMOS-VLC testbed uses an open and modular system architecture to integrate hardware and software components and tools to operate a satellite system simulator which can be later integrated into a mission operations setup for command scripting testing, and personnel training, mission rehearsals and anomaly resolution. The VLC communications system will be integrated into the COSMOS testbed setup with two main purposes 1) simulate the VLC system model with software-in-the-loop (SIL) 2) integrate the VLC system as hardware-in-the-loop (HIL) with the mission software using the COSMOS core software agents. Once the VLC system is integrated with the COSMOS testbed we are able to validate the VLC swarm communications system as part of the four major processes in mission operations, namely the real-time contact operations, mission operations analysis, anomaly resolution and mission planning and scheduling.

The Hawaii Space Flight Lab has a COSMOS Testbed setup with Hardware in the Loop using the Attitude Determination and Control Test Facility (see top right Fig. 6). The MotherSat hardware can be simulated with a complete set of hardware subsystems, in particular the 3-axis Attitude Control, including the 3 reaction wheels, 3 magnetic torque rods, magnetometers, sun sensor, nadir sensor, star tracker, etc. which can run on the air bearing embedded in the ADCS Test Facility. The three ChildSats are set up on the test bench on the side to the air bearing with a representative flight computer such as the Raspberry PI or ESP32. The MotherSat can be tested in an attitude dynamic environment on the air bearing, and we can stimulate the sensors with the ADCS Test Facility stimulators (Helmholtz cage for magnetic field stimulation, sun simulator, nadir simulator, and GPS simulator, HSFL is also in the process of acquiring a starfield stimulator for use with star trackers). The actuators can also be tested on the air bearing to validate the attitude control algorithms.

3 VLC MODEL AND SIMULATION WITH COSMOS

3.1 VLC ISC MBSE Capture

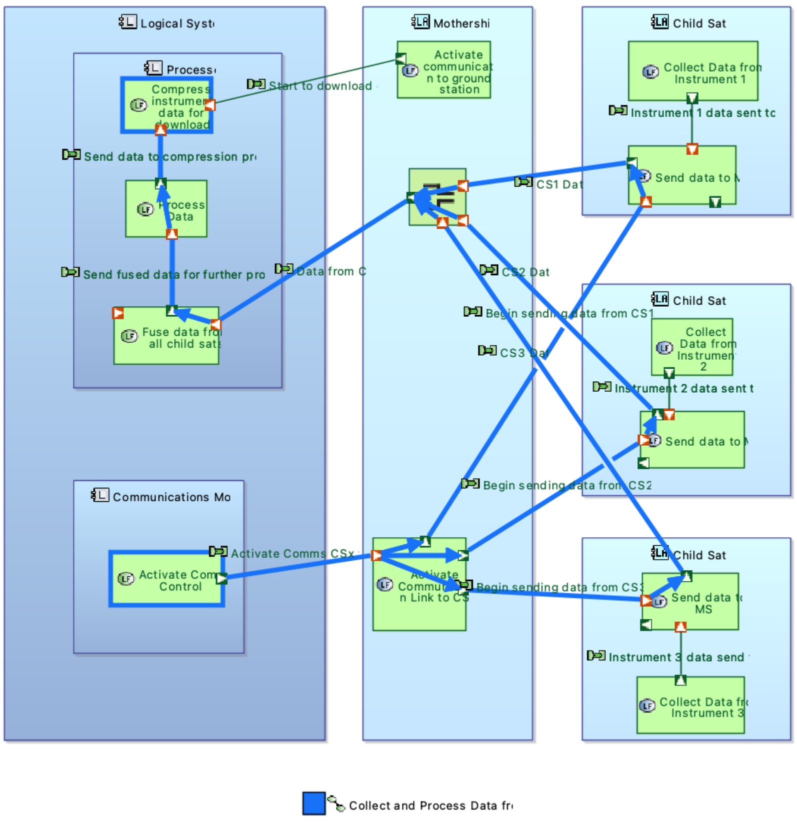

Logical Syst
Process
Compress instrumen data for download
Start to download
Send data to compression pr
Process Data
Send fused data for further pro
Fuse data fro all child sat
Communications Mo
Activate Com Control
Activate Comms CSx
Mothershi
Activate communicat n to ground station
Data from C
CS1 Dat
CS2 Dat
Begin sending data from CS1
CS3 Dat
Begin sending data from CS2
Activate Commun n Link to CS
Begin sending data from CS3
Child Sat
Collect Data from Instrument 1
Instrument 1 data sent to
Send data to M
Child Sat
Collect Data from Instrument 2
Instrument 2 data sent t
Send data to M
Child Sat
Send data to MS
Instrument 3 data send
Collect Data fr Instrument 3
Collect and Process Data fr

Fig. 7: Capella Logical Architecture of Swarm Communication with Process Flow

We have captured the logical and physical architecture of the bespoke system within the Arcadia/Capella MBSE environment with the process flow of the VLC ISC operations using the logical architecture diagram, see Figure 7. The logical architecture represents the logical components of the VLC ISC swarm testbed and does not reflect the technology that will be used.

The Capella Physical Architecture diagram of the bespoke system is shown in Fig. 8, and represents how the system is to be built. Also, captured in this diagram is the information on what physical components of what makes up the ISC subsystem on each satellite.

3.2 Analysis of Design

In integrating the VLC communication system into the COSMOS testbed will allow modeling and simulation to be performed for evaluating the performance and fine-tuning of the subsystem for inter-satellite communication. Of particular importance is to simulate the operational environment of space to best determine the design of the ISC system for mission assureness. Design factors based on distance between satellites and transmission power of LED, what are the constraints in terms of the following parameters:

- Continuous line of sight
- Size, configuration, and placement of transmitter LED(s)
- Size, configuration, and placement of receiver photo-diode(s)

We proposed to utilize I^{2}C as the interface between the VLC subsystem and the OBC running the COSMOS environment, which will allow testing of data bus < 1.5 Mhz.

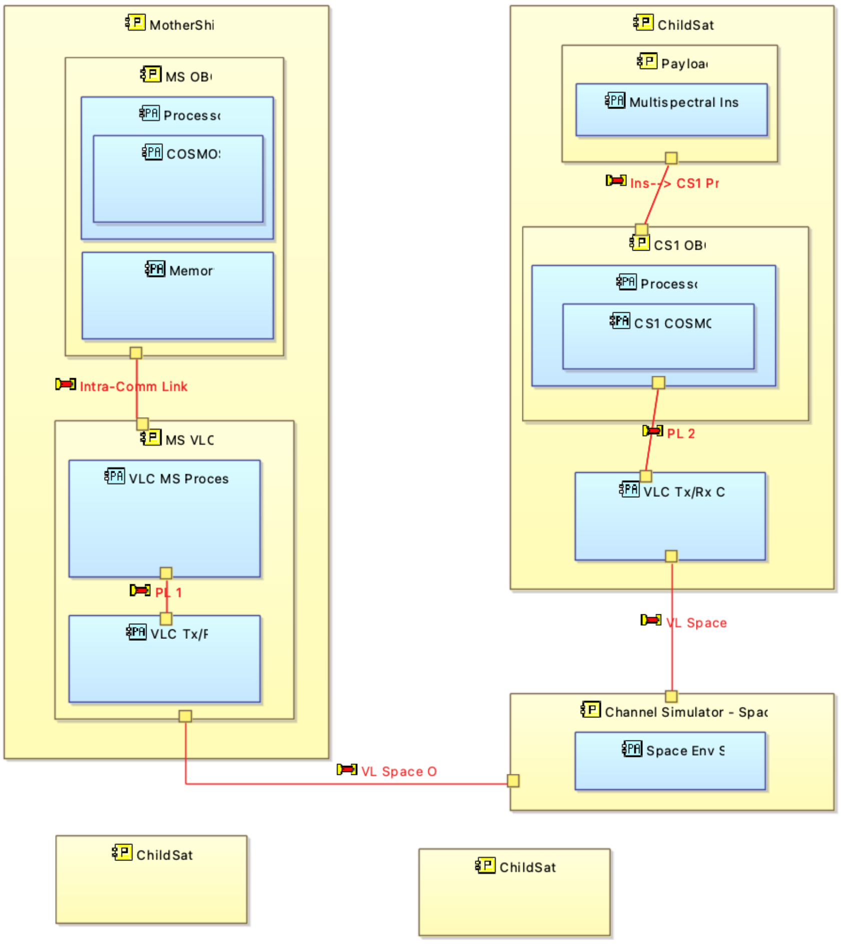


Fig. 8: Capella Physical Architecture of VLC - COSMOS system

4 VLC ISC TESTBED ARCHITECTURE

The test-bed architecture discussed will be captured in the model-based systems engineering (MBSE) tools of the open-source program Capella based on the Arcadia methodology. Captured will be the high-level testbed architecture based on the proposed concepts of operation. We will begin with operational analysis, then system analysis, followed by logical architecture, and finalize the MBSE process with a high-level view of the physical architecture. The main reasons for using the MBSE methodology is that it represents a single source of knowledge thereby reducing the amount of documentation required to capture the various aspects of the systems engineering process. It also represents a common language that can be used across all the involved disciplines.

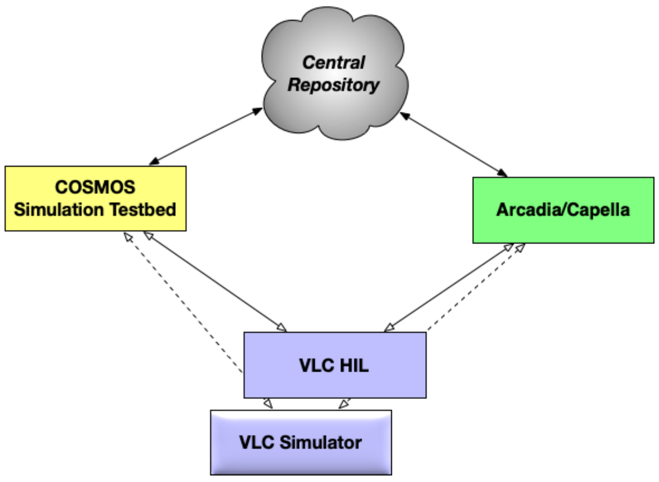


Fig. 9: General layout of VLC ISC Testbed

The general architecture for the proposed testbed that integrates both the modelling and simulation environment with the VLC system under test is shown in Fig. 9. The central repository represents where the COSMOS software and the Eclipse Arcadia/Capella, but also all of the information and data for the system under test resides including all requirements, design parameters from Capella, ephemeris and orbit propagation data, etc. The system under test that is modeled via Capella and connected to the COSMOS Simulation Testbed can either be a simulation model (VLC Simulator) and as the design progress to implement the actual hardware within the testbed environment as a hardware-in-the-loop (VLC HIL).

5 SIMULATION

We want to simulate the operability of an LED based visible light communication system for inter-satellite communications within a swarm constellation of small satellites composed of a MotherSat and 3 ChildSats. By integrating the orbital behaviour of each child sat within the swarm simulation will illustrate the design constraints of the VLC subsystem including optimization of physical location on each of the satellites. Other design decisions include:

- Intensity required to achieve appropriate SNR - Transmission Power
- Line of sight maximum angle of arrival - will affect design
    - Photo-diode array design of receiver
    - LED array design of transmitter
- Network layer OSI protocols from disruption of LOS due to orbital dynamics

COSMOS Simulation. COSMOS core includes a simulation environment with orbital dynamics, attitude dynamics and satellite subsystem simulation such as solar power generation, sensor optics, and others. In Fig. 10 shows orbital physics simulation architecture overview.

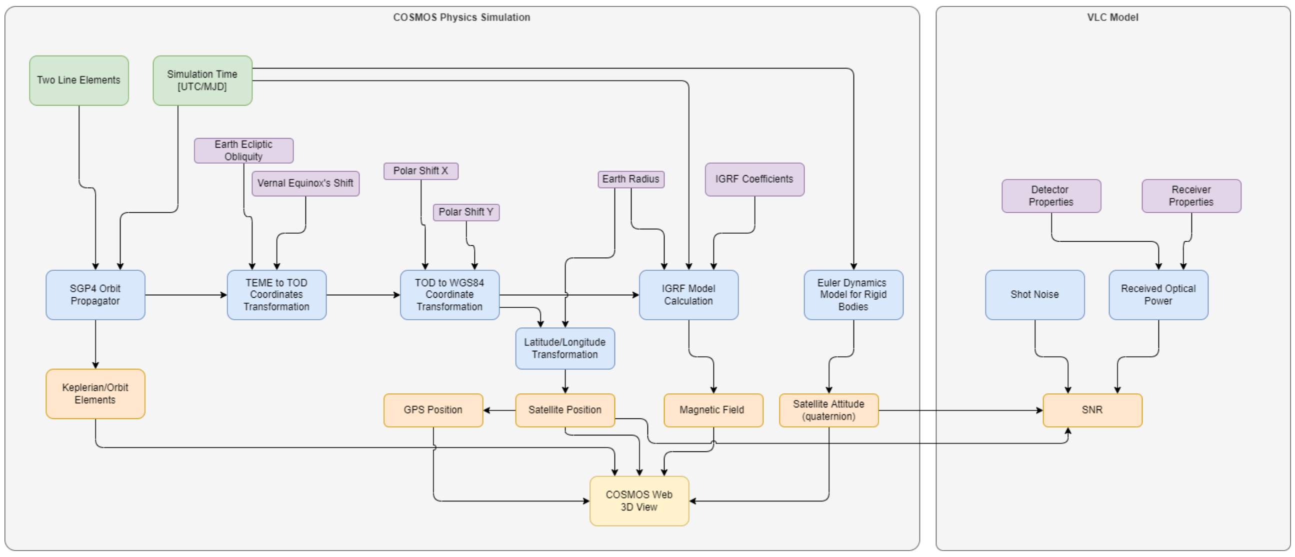


Fig. 10: COSMOS orbital physics simulation architecture overview

Two different simulation environments were developed, based on the propagation algorithm. For real-time and near real-time (up to 20x) a multi-agent approach was developed that allows the distribution of the simulation engine and nodes across multiple platforms. The different nodes communicate with each other, and with the simulation engine, over standard network sockets, much as they would in space. For high speed simulation, allowing us to perform Monte Carlo simulations, we developed a single program that contains both the simulation engine and nodal activity. All relevant information is then shared immediately in memory.

In each case we can initialise all relevant parameters through a JSON based command line string. The output from each run is collected into a uniquely named tab-separated data file.

COSMOS Visualisation / User Interface

COSMOS has an in-browser visualisation and commanding tool, COSMOS Web, to be used alongside COSMOS projects. An existing C++ prototype of the backend was rewritten in Javascript so that both server and client code would be in Javascript, providing a single language to aid developers to develop and for easier communications between the server and clients portions of the code. COSMOS Web, shown in Figures 11 and 12, has predefined data visualisation components that can be used as views for data. Examples of components are strip charts, tables, and 3D globes. Data is either fed into COSMOS Web in real-time as running COSMOS agents broadcast their definable state of health messages, or imported from a file. This data is in a JSON format and the basic COSMOS Web component accepts a key to access the JSON object to display. Users can create a layout file with various components with their data keys and size and positions on the page to display in the browser.

COSMOS Web is also a management tool and is able to send out agent requests or execute functions to running COSMOS agents. This can change the behaviour or state of an agent, have them output some specified data, among other capabilities. Components also accept functions to manipulate the data accessed by the provided key, so that data can be manipulated before visualizing. One example used in this project was passing in the Euclidean distance formula to plot the distance between nodes, where the raw data had provided positional values.

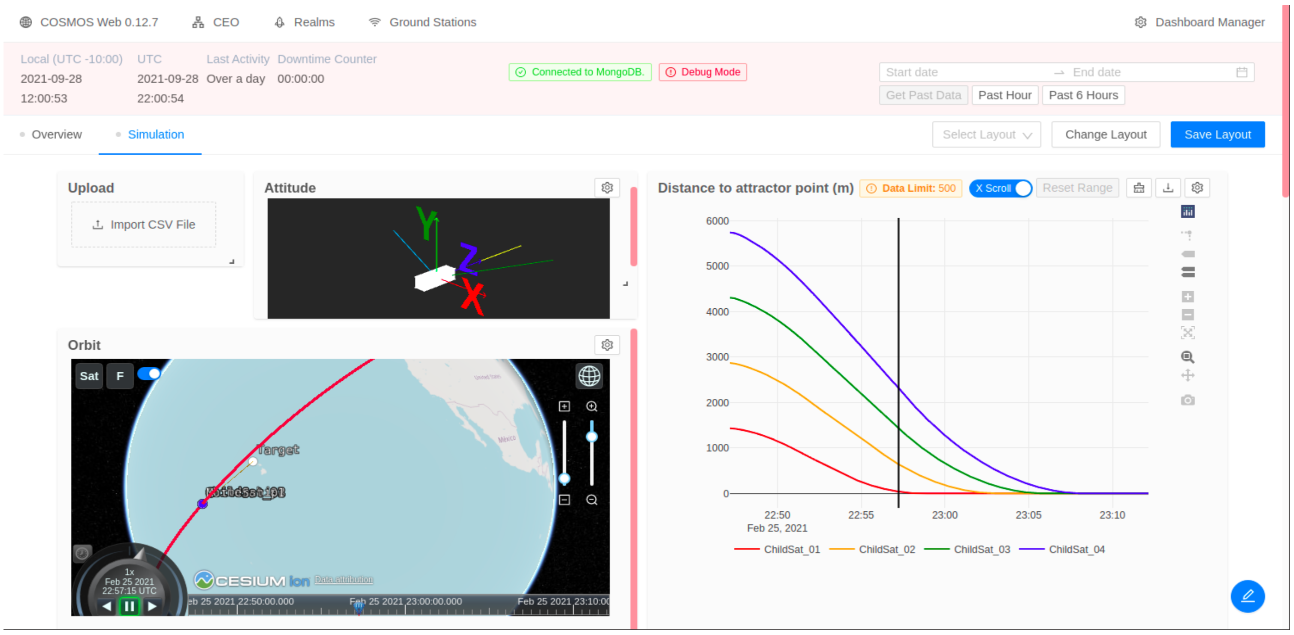
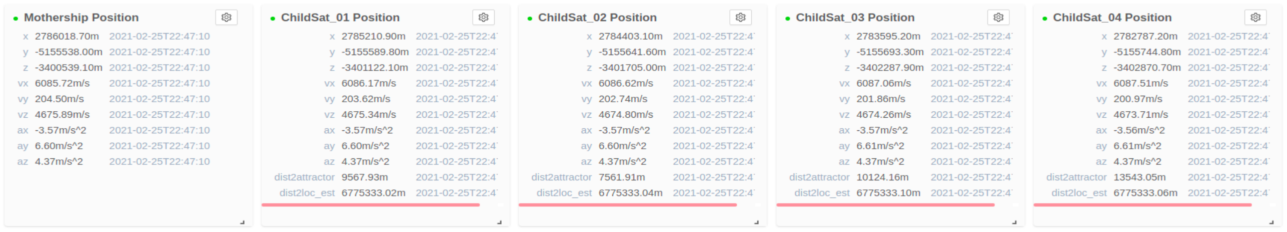


Figure11. An example layout page. Visible are the Attitude, Globe, and Strip Chart components. Timeline is synced across data views and playable at any speed.



Mothership Position

x	2786018.70m	2021-02-25T22:47:10
y	-5155538.00m	2021-02-25T22:47:10
z	-3400539.10m	2021-02-25T22:47:10
vx	6085.72m/s	2021-02-25T22:47:10
vy	204.50m/s	2021-02-25T22:47:10
vz	4675.89m/s	2021-02-25T22:47:10
ax	-3.57m/s^2	2021-02-25T22:47:10
ay	6.60m/s^2	2021-02-25T22:47:10
az	4.37m/s^2	2021-02-25T22:47:10

ChildSat_01 Position

x	2785210.90m	2021-02-25T22:4
y	-5155589.80m	2021-02-25T22:4
z	-3401122.10m	2021-02-25T22:4
vx	6086.17m/s	2021-02-25T22:4
vy	203.62m/s	2021-02-25T22:4
vz	4675.34m/s	2021-02-25T22:4
ax	-3.57m/s^2	2021-02-25T22:4
ay	6.60m/s^2	2021-02-25T22:4
az	4.37m/s^2	2021-02-25T22:4
dist2attractor	9567.93m	2021-02-25T22:4
dist2loc_est	6775333.02m	2021-02-25T22:4

ChildSat_02 Position

x	2784403.10m	2021-02-25T22:4
y	-5155641.60m	2021-02-25T22:4
z	-3401705.00m	2021-02-25T22:4
vx	6086.62m/s	2021-02-25T22:4
vy	202.74m/s	2021-02-25T22:4
vz	4674.80m/s	2021-02-25T22:4
ax	-3.57m/s^2	2021-02-25T22:4
ay	6.60m/s^2	2021-02-25T22:4
az	4.37m/s^2	2021-02-25T22:4
dist2attractor	7561.91m	2021-02-25T22:4
dist2loc_est	6775333.04m	2021-02-25T22:4

ChildSat_03 Position

x	2783595.20m	2021-02-25T22:4
y	-5155693.30m	2021-02-25T22:4
z	-3402287.90m	2021-02-25T22:4
vx	6087.06m/s	2021-02-25T22:4
vy	201.86m/s	2021-02-25T22:4
vz	4674.26m/s	2021-02-25T22:4
ax	-3.57m/s^2	2021-02-25T22:4
ay	6.61m/s^2	2021-02-25T22:4
az	4.37m/s^2	2021-02-25T22:4
dist2attractor	10124.16m	2021-02-25T22:4
dist2loc_est	6775333.10m	2021-02-25T22:4

ChildSat_04 Position

x	2782787.20m	2021-02-25T22:4
y	-5155744.80m	2021-02-25T22:4
z	-3402870.70m	2021-02-25T22:4
vx	6087.51m/s	2021-02-25T22:4
vy	200.97m/s	2021-02-25T22:4
vz	4673.71m/s	2021-02-25T22:4
ax	-3.56m/s^2	2021-02-25T22:4
ay	6.61m/s^2	2021-02-25T22:4
az	4.37m/s^2	2021-02-25T22:4
dist2attractor	13543.05m	2021-02-25T22:4
dist2loc_est	6775333.06m	2021-02-25T22:4

Fig. 12: Tabulated data taken from JSON-format COSMOS agent outputs.

The Globe component displays a 3D model of the Earth, with models of the satellites of the swarm in orbit around it. The user can toggle the travelled path, sensor cones, and vectors indicating the acceleration and target location on and off. The attractor point used by the algorithm is also displayed. Some swarm control functionalities were also implemented to control the distance between the nodes and also the shape of their formation. When a data file is imported, all components on the page synchronize their timeline to the timeline of the globe component, allowing forward or backward playback of the historical data at any speed, as well as skipping around to any time.

The connections and integration of the VLC as a SIL with COSMOS is demonstrated. Figure 13 shows the SNR results for two orbits for a MotherShip and ChildSat initially separated approximately 10 meters apart. The VLC model was implemented as a COSMOS module using the orbital simulation for a realistic separation of the CubeSats. The next steps are to implement the attitude simulation with the VLC model. The results show a strong SNR (88 to 92 dB) with the relatively close proximity of the satellites. The attitude simulation will show a decrease of the signal when the satellite attitude moves away from the target receiver.

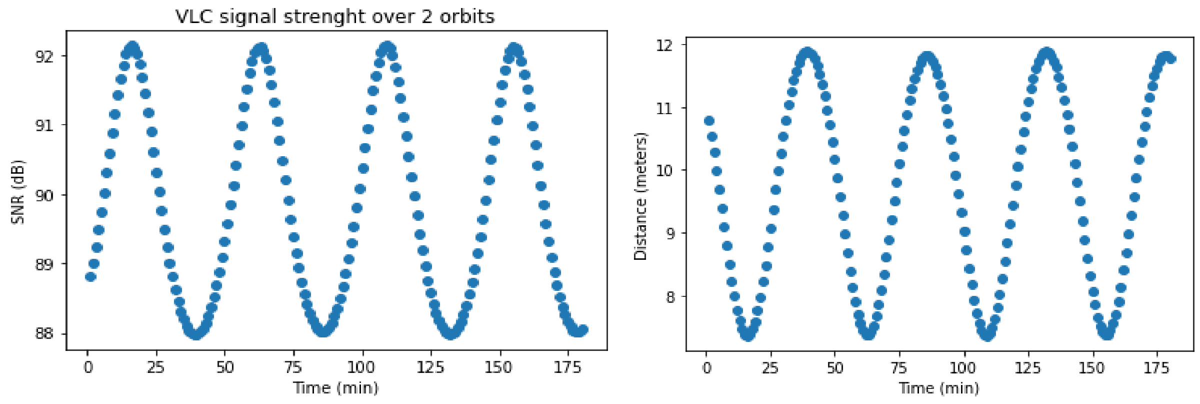


Fig. 13: VLC receiver over two orbits simulated with the COSMOS Physics engine a) Signal to noise ratio and b) Range Variability between CubeSat1 and the MotherSat

6 CONCLUSION

In this paper we propose a simulated testbed of a visual light communication system for resilient intersallite communications by looking at a cubesat swarm application with a mothersat and 3 childsats. We ran a preliminary simulation of the VLC model with the COSMOS physics engine to demonstrate the connection between the simulation environment and the Capella MBSE tool. The continued development of this project will be done with the Arcadia methodology using the Cappela software interface and COSMOS for the simulation environment. The next step is to implement the hardware in the loop system for validation with a representative cubesat swarm simulator.

7 REFERENCES

1. Alanazi, Abdulaziz, and Jeremy Straub. "Statistical analysis of cubesat mission failure." (2018).
2. Senge, Peter M. *The fifth discipline: The art and practice of the learning organization*. Currency, 2006.
3. Anyanhun, Awele I., and William W. Edmonson. "Inter-satellite communication MBSE design framework for small satellites." *2017 Annual IEEE International Systems Conference (SysCon)*. IEEE, 2017.
4. K. Cui *et al.* Line-of-sight visible light communication system design and demonstration. In *Proc. Commun. Syst. Netw. Digit. Signal Process.*, 2010.
5. Conney, Michael. "DARPA seeks high-speed inter-satellite communication technology." *Network World* (2015).
6. Friedenthal, Sanford, Alan Moore, and Rick Steiner. *A practical guide to SysML: the systems modeling language*. Morgan Kaufmann, 2014.
7. Haskins, Cecilia, et al. "Systems engineering handbook." *INCOSE*. Vol. 9. 2006.
8. L. Wood, W. Ivancic, and K. Dorpelkus 738 Using light-emitting diodes for intersatellite links, In Proc. IEEE Aerosp. Conf., Big Sky,MT, USA, 2010, pp. 1–6.
9. Amanor, David N., William W. Edmonson, and Fatemeh Afghah. "Intersatellite communication system based on visible light." *IEEE Transactions on Aerospace and Electronic Systems* 54.6 (2018): 2888-2899.
10. Rajagopal, Sridhar, Richard D. Roberts, and Sang-Kyu Lim. "IEEE 802.15. 7 visible light communication: modulation schemes and dimming support." *IEEE Communications Magazine* 50.3 (2012): 72-82.

11. Radhakrishnan, Radhika, et al. "Survey of inter-satellite communication for small satellite systems: Physical layer to network layer view." *IEEE Communications Surveys & Tutorials* 18.4 (2016): 2442-2473.
12. Roques, Pascal. *Systems architecture modeling with the Arcadia method: a practical guide to Capella*. Elsevier, 2017.
13. INCOSE, T. "Systems engineering vision 2020." *INCOSE, San Diego, CA, accessed Jan* 26 (2007): 2019.
14. NASA Technology Roadmaps TA 4: "Robotics and Autonomous Systems," 2015.
15. "Raspberry Pi Zero powers CubeSat space mission" 9th Mar 2022 https://www.raspberrypi.com/news/raspberry-pi-zero-powers-cubesat-space-mission/
16. National Aeronautics and Space Administration. *NASA Technology Roadmaps TA 4 : Robotics and Autonomous Systems*. (2015).
17. [REF.A.1] Space Studies Board, "Achieving Science with CubeSats: Thinking Inside the Box," National Academies of Sciences, Engineering, and Medicine, 2016,http://sites.nationalacademies.org/SSB/CompletedProjects/SSB_160539 https://www.nap.edu/catalog/23503 (2016) doi:10.17226/23503.
18. Sorensen, T., Pilger, E., Wood, M., Gregory, E. & Nunes, M. A. Development of the Mission Operations Support Tool (MOST). in *SpaceOps 2010 Conference Delivering on the Dream Hosted by NASA Marshall Space Flight Center and Organized by AIAA* (2010). doi:10.2514/6.2010-2230.
19. Sorensen, T. C., Pilger, E. J., Wood, M. S., Nunes, M. A. & Yost, B. D. Development of a Comprehensive Mission Operations System Designed to Operate Multiple Small Satellites. in *AIAA/USU Small Satellite Conference, Logan, UT. #SC11-IX-3* 1 (2011).
20. Sorensen, T. C., Pilger, E. J., Wood, M. S. & Nunes, M. A. Comprehensive Open-architecture Space Mission Operations System (COSMOS). in *American Institute of Aeronautics and Astronautics Plug-n-Play Mission Operations Workshop, May 16-17, 2011, San Jose, CA.* (2011).
21. Sorensen, T., Pilger, E., Yost, B., Nunes, M. & Differding, J. Plug and Play Mission Operations. *2012 IEEE Aerospace Conference* 1–13 (2012) doi:10.1109/AERO.2012.6187394.
22. Sorensen, T. C., Pilger, E. J., Wood, M. S. & Nunes, M. A. A University-developed Comprehensive Open-architecture Space Mission Operations System (COSMOS) to Operate Multiple Space Vehicles. in *SpaceOps 2012 Conference. Cleared for public release by NASA Ames* 1 (2012).
23. Nunes, M. A. *et al.* Expanding the Comprehensive Open-architecture Space Mission Operations System (COSMOS) for Integrated Guidance, Navigation and Control of Multiple Small Satellites. in *13th International Conference on Space Operations 2014* 1 (American Institute of Aeronautics and Astronautics, 2014). doi:10.2514/6.2014-1835.
24. Planetary Science Decadal Survey 2013-2022, https://solarsystem.nasa.gov/science-goals/about/
25. Astro2010: The Astronomy and Astrophysics Decadal Survey, https://science.nasa.gov/astrophysics/special-events/astro2010-astronomy-and-astrophysics-decadal-survey
26. Astro2020: Decadal Survey on Astronomy and Astrophysics 2020, https://www.nationalacademies.org/our-work/decadal-survey-on-astronomy-and-astrophysics-2020-astro2020
27. Decadal Survey for Earth Science and Applications from Space 2018, https://www.nationalacademies.org/our-work-decadal-survey-for-earth-science-and-applications-from-space