SSC24-P3-25

Loft Orbital's Approach to Rapid System Verification and Test

Paul Faugeras
Loft Orbital
Toulouse, France; +33 6 62 73 87 25
paul.faugeras@loftorbital.com

Matthew Hanley
Loft Orbital
Golden, Colorado; +1 231 286 3037
matt.hanley@loftorbital.com

Danielle Liut
Loft Orbital
San Francisco, California; +1 805 342 5435
danielle.liut@loftorbital.com

ABSTRACT

Loft Orbital's vision is to provide the fastest, simplest, and most reliable access to space for small satellite-class missions. Loft achieves this vision with a turnkey solution that implements the hardware and software solutions required to get a mission to orbit in months, not years. A key contributor to the success of Loft's vision is integrated systems testing in the hardware in the loop (HIL) environment.

Traditionally, HIL systems are designed and operated with one mission in mind. This means that for each new mission in which the architecture changes, a new HIL system needs to be created or an existing system needs to be altered. When resource contention inevitably occurs between users and begins to slow progress, the simplest resolution is to duplicate test setups to increase the quantity of test resources. This approach, however, is not ideal for speed or cost. Loft solves this problem by employing unique methods and technologies to reach an optimal solution.

Loft's HIL testing methodology differs significantly from the majority of the industry in two primary disciplines: automation and resource sharing.

This paper discusses the implementation of these technologies, other modern day DevOps practices, as well as their impact on software development efficiencies, ease of use, and system reliability. Employing these technologies allows Loft to react at an unprecedented speed to new missions and customers which ultimately provides a faster, more reliable, and simpler path to space.

INTRODUCTION

Loft Orbital is a small satellite-class company that deploys space infrastructure as a service with the goal of creating the fastest, simplest, and most reliable access to space. As with any satellite program, testing cannot be ignored. Traditionally, HIL systems such as FlatSats would be a large part of the schedule, budget, and therefore risk for a new program. Loft decided there must be a better way, and thus the "Test Infrastructure" team was created. The test infrastructure team at Loft is tasked with designing, building, maintaining, and automating the various HIL systems deployed at the company's three different locations: France, Colorado, and California. The test infrastructure team works towards Loft's goals by challenging assumptions made in the traditional aerospace environment and implementing new HIL technologies to optimize for the variables that Loft cares the most about: speed, simplicity, and reliability.

RESOURCE SHARING

Loft's space infrastructure services rely on the ability to share hardware resources, analogous to the resource sharing that is employed by cloud providers.[1] This inherently requires an abstraction layer to operate and automate a heterogeneous constellation of spacecrafts. In this context, just like abstraction layers have been designed and imple-

mented in Cockpit[2,3] (Loft's Automated Mission Control System), abstractions are also needed on Loft's Test Infrastructure in order for it to scale efficiently. Procuring new hardware, designing new harnesses, securing physical space, and finding the human-power to design, assemble, commission, and test a new HIL system is not condusive to Loft's ultimate vision.

Instead of deploying multiple traditional, static HIL systems each with dedicated resources,[4] Loft's test infrastructure team has adopted the concept of a "pool of resources." In such a pool, each individual piece of hardware is treated as its own entity. However, each unit is also able to by dynamically linked to any other unit within the ecosystem, thus allowing for the creation of brand new HIL systems on the fly.

Resource Pool Concept

The original idea comes from the concept of "Mankai" (Lemnoideae, a subfamily of flowering aquatic plants, also known as duckweeds, water lentils, or water lenses): the smallest possible component that is part of a larger pool of resources. Each component can be assigned to a community to fulfil a specific role and achieve a larger function in the ecosystem.

Loft achieves "Mankai" through the concept of The Hub.[5] The Hub is a hardware layer owned by Loft Orbital which connects the bus and the payloads. The functions of The Hub are to provide an abstraction layer between payloads and the bus, implement space-to-ground links, and provide compute and power resources to the payloads to meet their respective service level agreements.

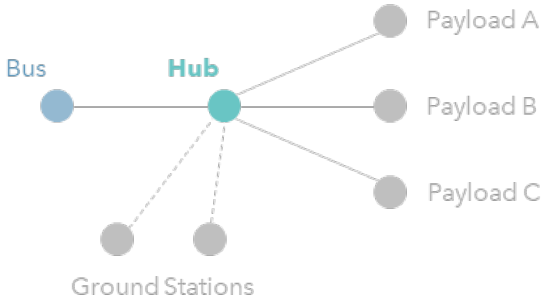

Figure 1: Simplified representation of Loft's hub architecture

All of the Loft's satellites have a version of the Hub, which is independent from and provides a layer between the bus and the payloads, as depicted in Figure 1. The Hub then becomes like the central nervous system of the satellite, and as such, the bus, payloads, and any other extremities can be treated as "yet another peripheral." This simplifies both the customers' and engineers' development flow as they do not need to concern themselves with the intricacies required to interface with various types of buses and payloads, they just have to understand how to use the central hardware abstraction interfaces exposed by the Hub.

"The Hub" itself is not just a single piece of hardware. It is made up of many different building blocks to create a system that meets the requirements for a given mission. Thus, it was necessary to separate it into multiple "Mankai" resources as well. A mission's Hub might, for example, include multiple compute elements, power distribution elements, space to ground links, and data storage elements. When designing a mission for a given customer, these elements can be combined in quantities that will meet the objectives of the mission. At a spacecraft level, two Loft vehicles may look quite different. But looking at them with a finer lens, one will find that each spacecraft is made up of many "Manaki" resources.

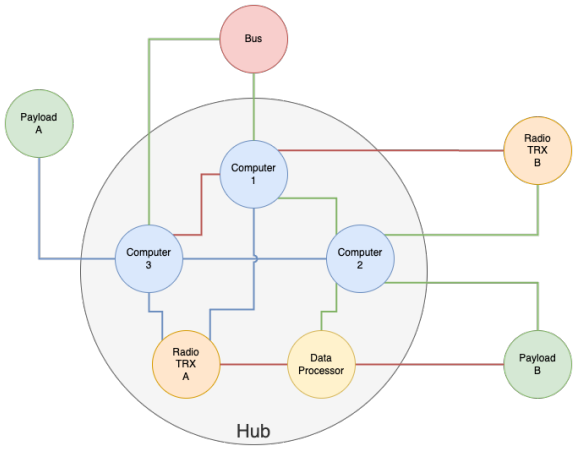

Figure 2: Simplified representation of Loft's Hub links architecture

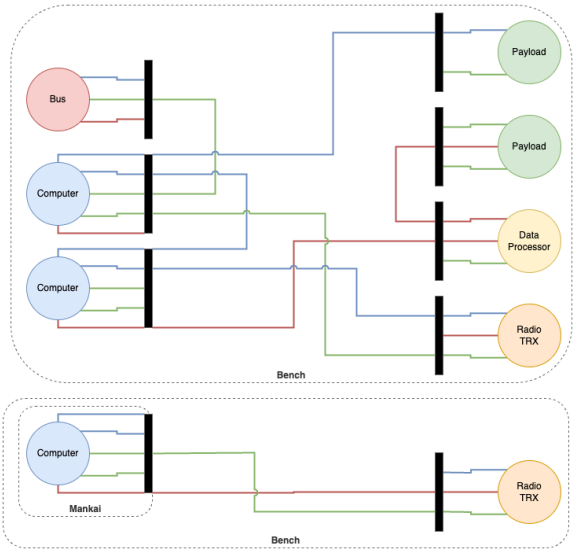

Figure 3: Mankai concept and visualisation on HIL systems

Link Virtualization

To achieve this concept of independent "Mankai" at the HIL system level, the test infrastructure team at Loft was tasked with devising a method to make each piece of hardware independent from each other piece, yet are able to be interfaced when needed. This led to two main objectives for the test infrastructure team when envisioning a new HIL. One requirement being that each node in the resource pool shall be remotely controllable, and the other being that each node shall be able to create a link between any of its interfaces and any other node's compatible interface(s). It is also highly desirable that interfaces are able to be monitored (i.e. adding a tap interface to a communication bus, or monitoring current flowing out of a power supply output) for traffic and behavior analysis, as well as debugging purposes.

The test infrastructure team performed trade studies for each interface type to determine how these requirements could be best met. It was determined that the objectives outlined above are achievable using commercially available lab and networking equipment.

Concerning "routable" signals, the chosen method was to generally encapsulate the sent data in TCP/IP packets, route the data, then de-encapsulate the packets and convert back to the original signal type. This does, however, introduce the notable drawbacks of bandwidth degradation, latency, and non-flight like electrical characteristics. Understanding and characterizing these drawbacks has allowed the engineers to accept these differences in a majority of the development environments in order to realize the benefit of the configurable HIL ecosystem. In the event that these drawbacks are not acceptable, Loft provides HIL systems that are more analogous to traditional FlatSats.

For other signals that are more challenging to encapsulate (analog signals, LVDS, etc.), or signals for which an encapsulation solution was not available commercially off-the-shelf, a low-level local routing solution was chosen, which, despite limiting the possibilities of geolocation distribution of the hardware resources, allows the device interface communications to be routed locally with low human and financial overhead.

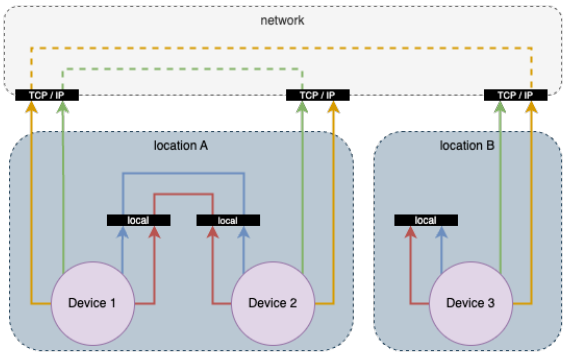

Figure 4: Local and remote links capabilities

Hardware Virtualization

In some cases, based on available resources and required setup, the minimal requirements might not be met to provide basic functionality for the user. For example, a user needing to test a flight procedure through the Mission Control System to operate a payload might find themselves in need of a bus, a specific component of the Hub, and a payload.

In this case, the minimal required setup for the user would consist of a computer running the Mission Control System, the bus, the payload, and the Hub element. Without this minimum set of resources, the procedure would not be able to be tested. Unforutnately, Loft does not own an infinite set of hardware resources. This means that the situation arises that one of the required hardware dependencies is not available. To combat this, "hybrid" systems can be deployed where part of the setup can be emulated through the help of one or more "virtual hardware resources." A hybrid system then consists of traditional HIL elements, but can also make use of virtualized hardware running locally, on-premise, or even in the cloud. This virtulaized hardware can

make use of the "Link Virtualization" discussed before to route data which can then be converted to a desired phyiscal protocol.

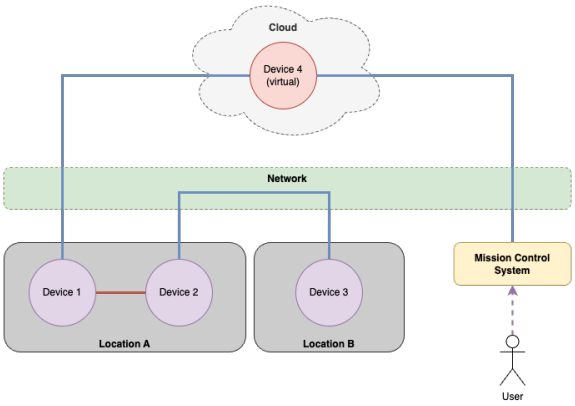

Figure 5: Hybrid HIL/SIL capabilities

AUTOMATION

The HIL automation consists of three major parts: Infrastructure as Code, Resource Availability Solving, and System Deployment. Each of these parts plays a role in allowing the HIL ecosystem to map hardware in the resource pool to the requirements of a mission then deploy in a manner that can be utilized for development and test.

Infrastructure as Code

In order to automate deployment of a spacecraft configuration to the HIL ecosystem, Loft's automation system needs to know what pieces of hardware are available and compatible for use, what role each piece will have in the setup, and how the hardware is interconnected.

In order to implement this, Loft leveraged various Machine-to-Machine (M2M) languages that allow the infrastructure automation to:

1. identify which resource is available, and which is not (schedule management),
2. retrieve the configuration of the satellite (desired state), and what role each resource will play in this configuration.
3. understand how each interface of each resource is reachable/routable,

Using these three pieces of information, the system is then able to de-conflict between the resources the user has access to, their available interfaces, and the desired setup (what satellite program they are looking to replicate), in order to build a deployment plan that is "as flight representative as possible" to the required state.

Concerning the scheduling of resources, resource availabilities are stored in a real-time database, as these are likely to change often (i.e. every time a user reserves or releases a resource). (read below, *Resource Availability Solving*).

Many of Loft's hardware resources can be configured to perform one of many roles at the time of software and firmware installation. These different roles are all recorded in data files describing each resource. These data files are referenced to determine if a given resource is compatible with a given mission when deploying a specific setup. Also recorded in these data files is information about interfaces. For the automation system to understand if two units in the resource pool can be connected, information about each interface is encoded its respective data file. For example, it should not be allowed that a RS-422 interface is connected to a SpaceWire interface, but it should be allowed that two or more CAN buses are able to be connected.

Resource Availability Solving

The available resources are stored in a real-time database, as opposed to their interfaces and the satellite architectures, which are both stored in a configuration data storage. All three data sources being only accessible from within the Loft infrastructure.

Through a booking system, users can specify what minimal resource they need for what purpose, and the software will inform them on what is available that can answer their needs, and create a booking, then update the schedule so that future users can not double-book the same resource.

Regarding the deconfliction of states, Loft developed internal tools, accessible to developers, to provide them with the best possible solution to their requirements, with the resources they have access to.

For example, if a developer requires testing a certain feature on a specific subset of the Hub components, they can specify it in their requirements, and not immobilize resources that could otherwise be used by another developer on a different test setup.

System Deployment

Setup Deployment is the part of the automation that combines the "infrastructure as code" with the

"resource availability solving" and actually deploys a physical setup with the desired configuration. This is coordinated using a Direct Acyclic Graph (DAG) workflow execution:

1. booked resources and their interfaces are fetched, and required satellite program configuration–hardware and software–is pulled,
2. a "development" setup is put into place, that is deterministic and independent from the selected satellite program, and which the HIL automation will recognize and use to execute the next step,
3. each piece of hardware is deployed with its required software and firmware configuration from this automation,
4. all surrounding electrical ground support equipment (EGSE) software is deployed,
5. a link virtualization map is built with the conflict resolver which defines how the various pieces of the resource pool will be connected,
6. the virtualized links are made using the various lab and networking equipment built into the HIL ecosystem.

Once all steps above have been executed, either the test setup is transferred to the user for manual testing/development through the Mission Control System, or automated testing is executed and test results are tracked using Loft's in-house test tracking system.

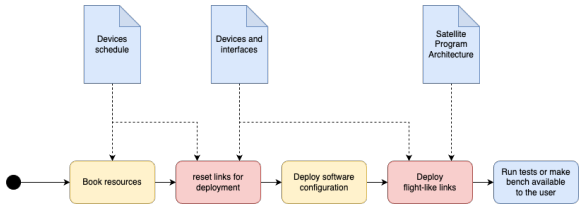

Figure 6: Deployment Workflow DAG

IMPACT ON DEVELOPMENT AND OPERATIONS

Without human in the loop interaction, this new automation system has allowed Loft's developers to access a **reproducible**, **representative**, and **fail-safe** testing environment that is compatible with any satellite program, without needing to scale test infrastructure hardware needs or team size.

It has also allowed Loft to quickly and easily prototype architecture changes, troubleshoot software and communication protocol issues faced in assembly, integration and testing of spacecraft in a matter of minutes, instead of hours or days traditionally required.

This, in turn, has allowed employees from any background (technical or not) to get familiar with the system architecture of Loft's satellites in the context of training and execution of Loft's concept of SatDevOps[6] (Satellite Development and Operations).

Finally, it also has allowed Loft to much better track requirements coverage, software feature progression, and automate warning systems for developers to know when a regression is introduced in the codebase, by putting into place nightly tests, and longer form endurance tests.

The implementation of such a system has allowed the test infrastructure team greatly accelerate the HIL testing process. New missions can be brought up in a matter of hours instead of weeks. New payloads can be integrated into the ecosystem with simple harnessing and a configuration file. Old missions can continue to be supported without monopolizing physical space while sharing their resources with future missions. All of this makes for a much faster, simpler, and more reliable path to space.

FUTURE IMPROVEMENTS

As explained in this paper, some communication protocols are quite challenging to encapsulate into TCP/IP frames with affordable COTS components, and hence require local electronic matrix routers. This is great in regards to performance (the routing being low-level, it does not add abstraction layers), but quite limiting in terms of route determination and deployment, since it does not fully leverage the TCP/IP protocol advantages. This also limits Loft's ability to deploy multi-site test setups (where part of the hardware pieces are running in one location, and others are running in another location).

Finally, the hybrid HIL/SIL capabilities described in this paper currently only support a subset of the hardware required for Loft's missions. The team is currently working on adding the capability to virtualize additional Hub hardware components as well payloads to allow a range of test environments that vary from fully hardware based to fully software based, and all the combinations in between.

References

[1] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, and Matei Zaharia. A view of cloud computing. *Commun. ACM*, 53(4):50–58, apr 2010.

[2] Lucas Brémond, Etienne Vincent, Gauthier Damien, Caleb MacLachlan, and Brunston Poon. Lessons learned from automating heterogeneous spacecraft systems. *17th International Conference on Space Operations*, 2023.

[3] Rémy Derollez, Robin Petitdemange, and Lucas Brémond. Automated planning and scheduling system for a heterogeneous spacecraft constellation. *17th International Conference on Space Operations*, 2023.

[4] Tanin Saroj, Suwat Sreesawet, Sittiporn Channumsin, Nisarat Kaewkhuntod, and Pichaya Tandayya. Hardware-in-the-loop simulation for satellite system verification. *International Electrical Engineering Congress*, 2022.

[5] Loft Orbital. Our technologies: the universal interface payload hub. *https://www.loftorbital.com/technology*, 2024.

[6] Brunston Poon, Lucas Brémond, Caleb MacLachlan, and Leon Stepan. Satdevops: A novel automated satellite operations methodology. *17th International Conference on Space Operations*, 2023.