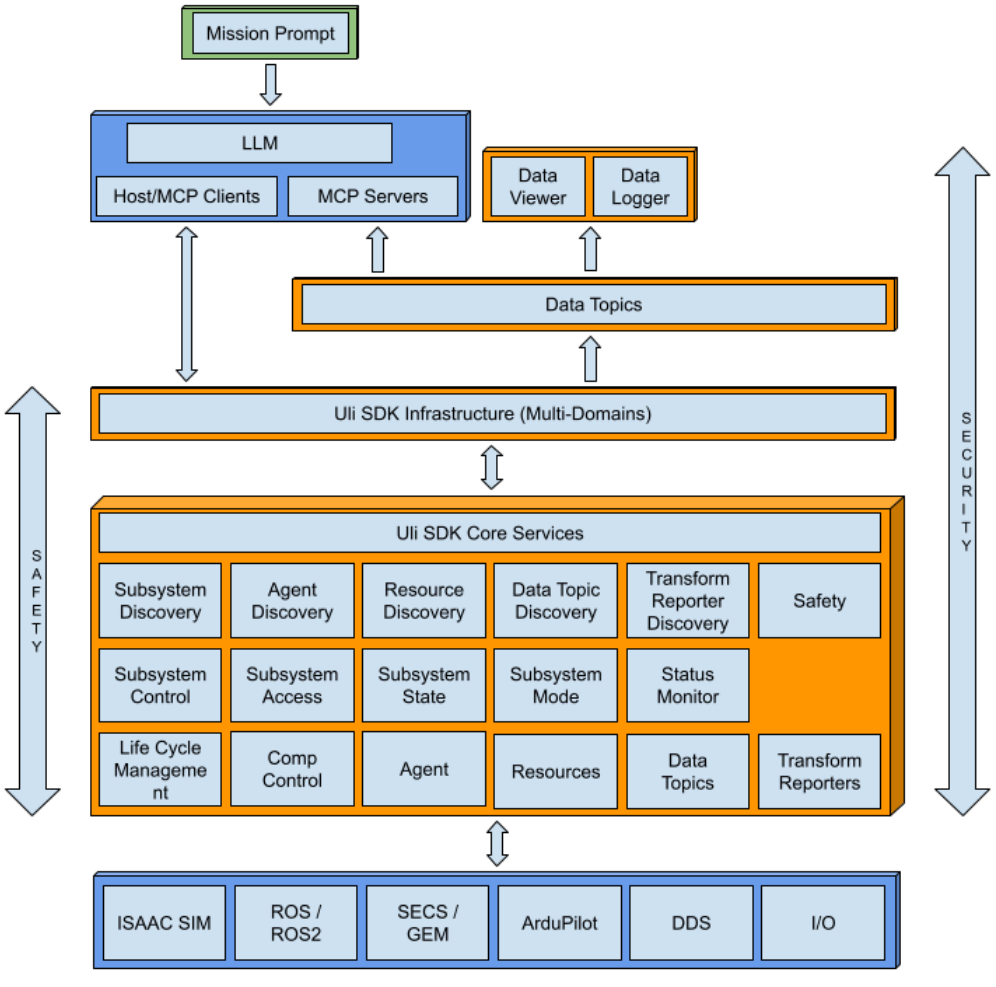
ULISDK

Bridging AI with Multi-Domain Data & Controls

The **Uli (Unified Link Interface) SDK** comprises a comprehensive set of software libraries developed to construct architectures for applications across multiple domains.

By offering standardized, unified interfaces, and dynamic functional modules within the architecture framework, the Uli SDK aligns with the Department of Defense (DoD) **Modular Open Systems Approach (MOSA)**. This ensures smooth interoperability between modules, supporting the dynamic addition, removal, or reconfiguration of components as operational requirements evolve.

Building on its foundational interoperability, the Uli SDK empowers advanced **Data Visualization** and **AI agent integration** by providing consistent interfaces to access and interact with subsystems, agents, data topics, resources, and transformations. This enables developers to create sophisticated solutions that enhance situational awareness, optimize decision-making, and improve system performance. See the block diagram below:



Note: Uli SDK Implements the orange blocks

Figure 1 Block Diagram

Infrastructure

In today's interconnected world of cloud, servers, and devices, Uli SDK organizes computing resources into logical subsystems and nodes. As depicted in the diagram below, a subsystem comprises one or more nodes, and each node hosts one or more computational applications (comp apps) that implement related services. These services communicate with one another using message-based interactions.

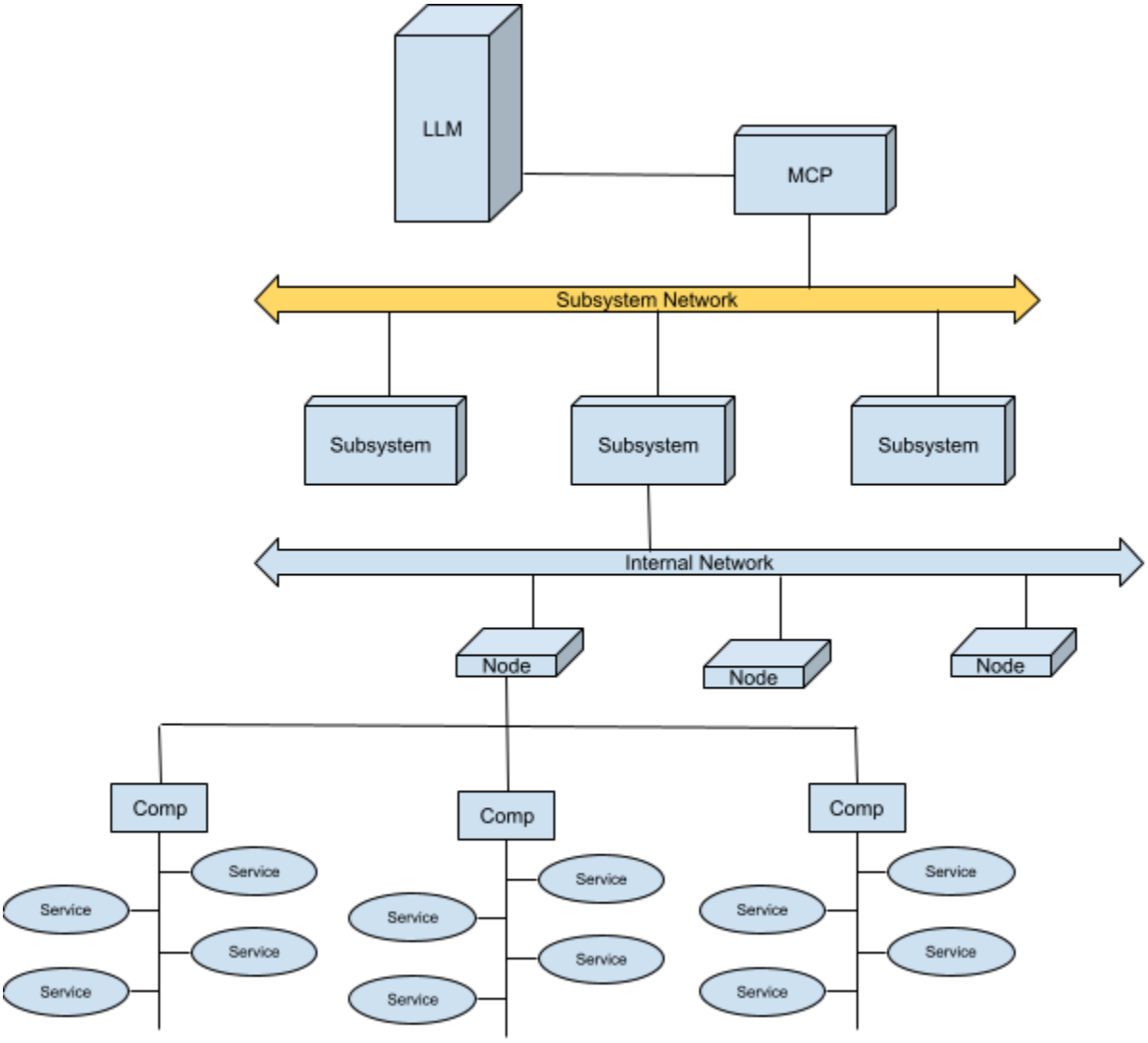


Figure 2 Network Infrastructure

Subsystems, nodes, and comp apps are each assigned unique identifiers within their respective contexts. Messages exchanged between servers and clients are addressed using tuples in the format (Subsystem ID, Node ID, Comp ID), which specify both the source and destination of the communication.

The Subsystem ID and Node ID can be dynamically assigned. Similar to how a DHCP server assigns IP addresses, the Uli SDK's Id Allocator service assigns unique IDs to requesting subsystems. Once a Subsystem ID is assigned, the Subsystem Manager service within that subsystem allocates IDs to the requesting nodes.

Messages are serialized for efficiency using Cap'n Proto (see capnproto.org). The serialized messages communicate between servers and clients via local socket and UDP unicast and multicast packets. Messages can also be assigned to specific partitions so that only the servers and clients of the same partition can communicate.

Each node implements a Node Manager service that handles message routing. It dynamically learns routing information from the messages it processes, enabling it to forward messages to their destinations without requiring IP addresses for communication.

Services

For each service, Uli SDK defines and implements input/output messages, internal events (events within the comp app), a state machine, and configurable parameters. Related services are grouped together into computational applications (Comp Apps).

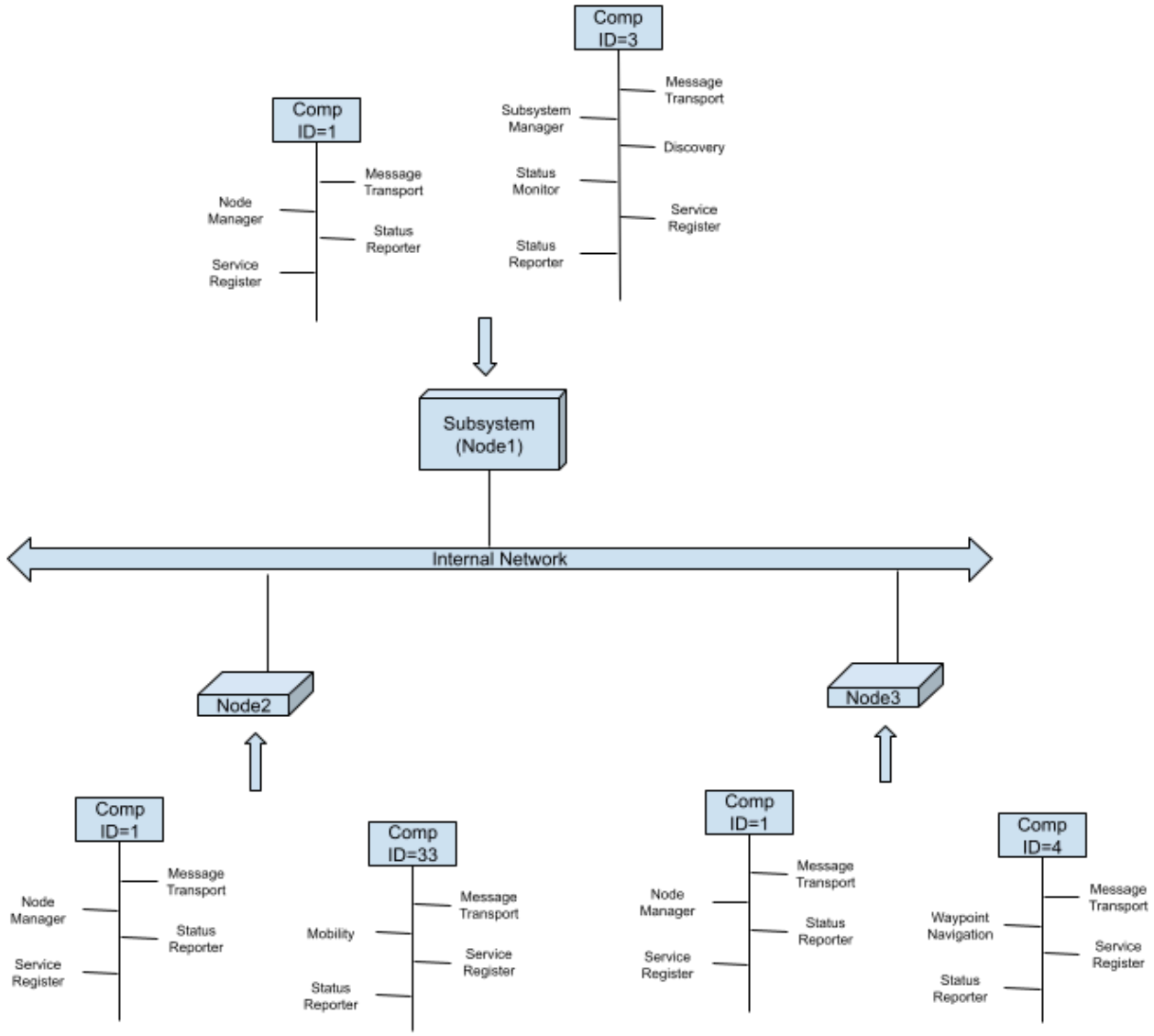


Figure 3 Services and Comp Apps in a Subsystem of three nodes

Each Comp App must include the Service Register and Status Reporter services. The Service Register enables the discovery of services within the Comp App, while the Status Reporter provides detailed information on the status of those services.

Resources

Resources are digital entities within the subsystem, encompassing video sources, web sockets, drivers, manipulators, machine learning (ML) models, and meta characters. Clients can access these resources via the Resource Discovery service.

Agents

Agents are computing modules that perform mission-critical tasks. They are discoverable via the subsystem's Agent Discovery service, allowing clients to query each agent's capabilities, configurations, and status, and to request exclusive control over them.

Agents serve as essential resources for AI Agents within LLM frameworks. They provide the tools necessary for task execution, deliver capability and mission-readiness data that support AI-driven workflow reasoning, and supply control parameters and feedback to help AI Agents learn and optimize processes.

Data Topics

Data Topics refer to the categories of data published by the services. These can be discovered through the Data Topic Discovery service of the subsystem. Clients can subscribe to the Data Topics of interest.

Transforms

Transforms represent the quaternion conversion between two coordinate frames. They are provided by the Transform Reporter services, which can be located via the Transform Discovery service of the subsystem. Clients can query the Transform Reporter services to obtain the transform between specific coordinate frames.

Mission Execution

The mission is carried out through agents that perform essential tasks. The Uli SDK offers a unified interface for discovering, configuring, executing, and monitoring the status of these agents.

Data Access

The Uli SDK supports Data Topic Discovery, allowing applications within the subsystem to register their published data topics. Through a unified interface, clients can discover and subscribe to these data topics.

Data topics can be visualized and logged using our Data Viewer and Data Logger, or fed to MCP services for LLM.

Safety

The Uli SDK includes a Safety Service that monitors user-initiated emergency stop (e-stop) events and the health status of services within applications of the subsystem. When an emergency condition is detected, it is escalated and propagated to safety-critical functions.

Additionally, application message transports can be partitioned to separate communications between safety-critical and general-purpose functions, enhancing the overall safety of the system.

Security

To safeguard against cyberattacks, the Uli SDK employs secure messaging and authentication between services and clients. For inter-node communications, it uses Datagram Transport Layer Security (DTLS) to ensure secure data transmission. For mission control and data access, certificate-based authentication is used to verify clients and manage their access permissions.

Integration

The Uli SDK automatically generates Python bindings for applications, facilitating seamless integration within the PySide6, Flask-Next.js and MCP frameworks, as well as NVIDIA Isaac Sim, ROS, SECS/GEM, DDS, and a diverse array of hardware and sensors.

Development

Uli SDK develops applications for:

Devices – X86_64
Nvidia Jetson Nano, Xavier NX, Orin Nano, NX, AGX

Operating Systems – Ubuntu 18.04, 20.04, 22.04, 24.04
Nvidia Jetpack 4.6, 5.1, 6.0

Tools

The Uli SDK offers a comprehensive suite of tools for code generation, building, and application deployment.

The SDK's code generation tools produce concise C++ code for record structures, messages, services, and Comp Apps. For example, the Uli-generated service code typically contains around 400 source lines of code (SLOC).

Uli SDK utilizes Google's highly efficient build tool, **Bazel**, which supports cross-building for both x86 and ARM 64-bit architectures, with or without CUDA support. It has been tested on Ubuntu 18.04, 20.04, 22.04, and 24.04, and supports x86_64, NVIDIA Jetson, Xavier, and AGX boards. Here are the Uli SDK supported OS and devices:

OS	Device	Development Host (x86_64)
Jetpack 4.6	Jetson Nano, Jetson Xavier NX	Ubuntu 18.04
Jetpack 5.1	Jetson Xavier NX, Jetson Orin	Ubuntu 20.04
Jetpack 6.0	Jetson Orin Nano, AGX Orin	Ubuntu 22.04
Ubuntu 18.04	x86_64	Ubuntu 18.04
Ubuntu 20.04	x86_64	Ubuntu 20.04
Ubuntu 22.04	x86_64	Ubuntu 22.04
Ubuntu 24.04	x86_64	Ubuntu 24.04

In addition, the SDK includes shell scripts to streamline the staging and deployment of applications across a network of devices.

Highlights

1. **Modular Open System Approach (MOSA)** provides interoperability and flexibility of the system architecture.
2. **Secure messaging** and mutual authentication between services and clients.
3. **Dynamic ID assignment** for assets, computing modules, and applications.
4. **Discovery** of assets, services, resources, data topics, agents, and transforms.
5. **Data access authorization** categorized into classified, controlled, and unclassified tiers.
6. **Exclusive service control**, ensuring controlled access to services.
7. **Lifecycle management** of services, including startup, runtime, and shutdown phases.
8. **Service health reporting and monitoring** to track application status.
9. **Emergency stop (e-stop) propagation** to allow services to manage critical situations.
10. **Support for various operational modes**, including Standard, Reduced, Training, Maintenance, and user-defined modes.
11. **Integration with** NVIDIA ISAAC Sim, ROS, SECS/GEM, DDS, and a wide range of hardware and sensors.
12. **Code generation tools** that produce C++ code for record structures, messages, services, and applications.
13. **Auto-generated Python Bindings** that facilitate seamless integration within the MCP and Flask-Next.js backend frameworks.
14. **Cross-build support** for x86_64, NVIDIA Jetson Nano, Xavier, and AGX, as well as multiple Ubuntu versions (18.04, 20.04, 22.04, 24.04, Jetpack 4.6, 5.1, 6.0).
15. **Comprehensive examples** of applications, data viewers, data loggers and QT PySide6 UI.

Advantages

The Uli SDK stands out for its secure and dynamic features, providing key advantages:

Safety: Monitoring the user e-stop event and the health conditions of the services. Partition message transports between safety-critical and general-purpose functions.

Security: Secure messaging and mutual authentication between services and clients.

Unified agent and data access interfaces: enable the discovery of agents and data topics, and support agent execution and data topic subscriptions. Data access is categorized into classified, controlled, and unclassified levels, with separate storage for each.

Quality: Quality assurance is embedded throughout the development lifecycle, with continuous review and validation to ensure objectives are met.

Cost-effectiveness: The Uli SDK is open-source for licensed customers, offering low overall costs.

Visit our website: www.ulisdk.com