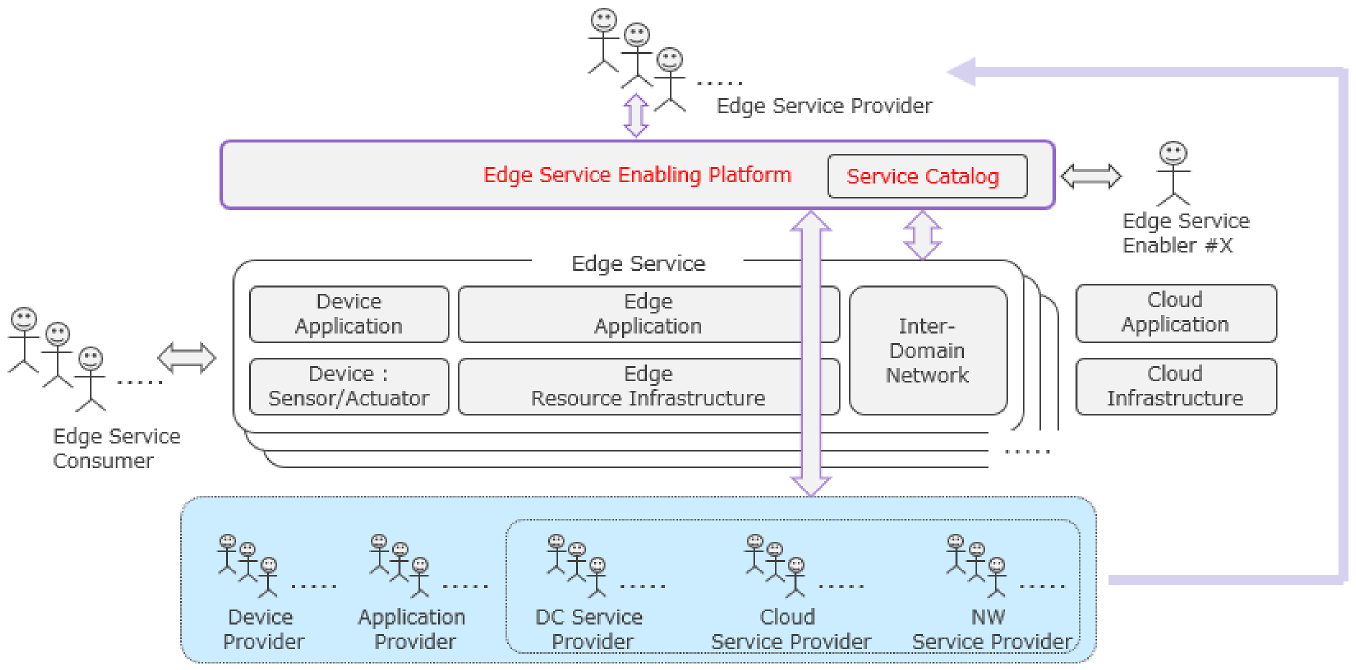
Edge Service Enabling Platform Blueprint Architecture

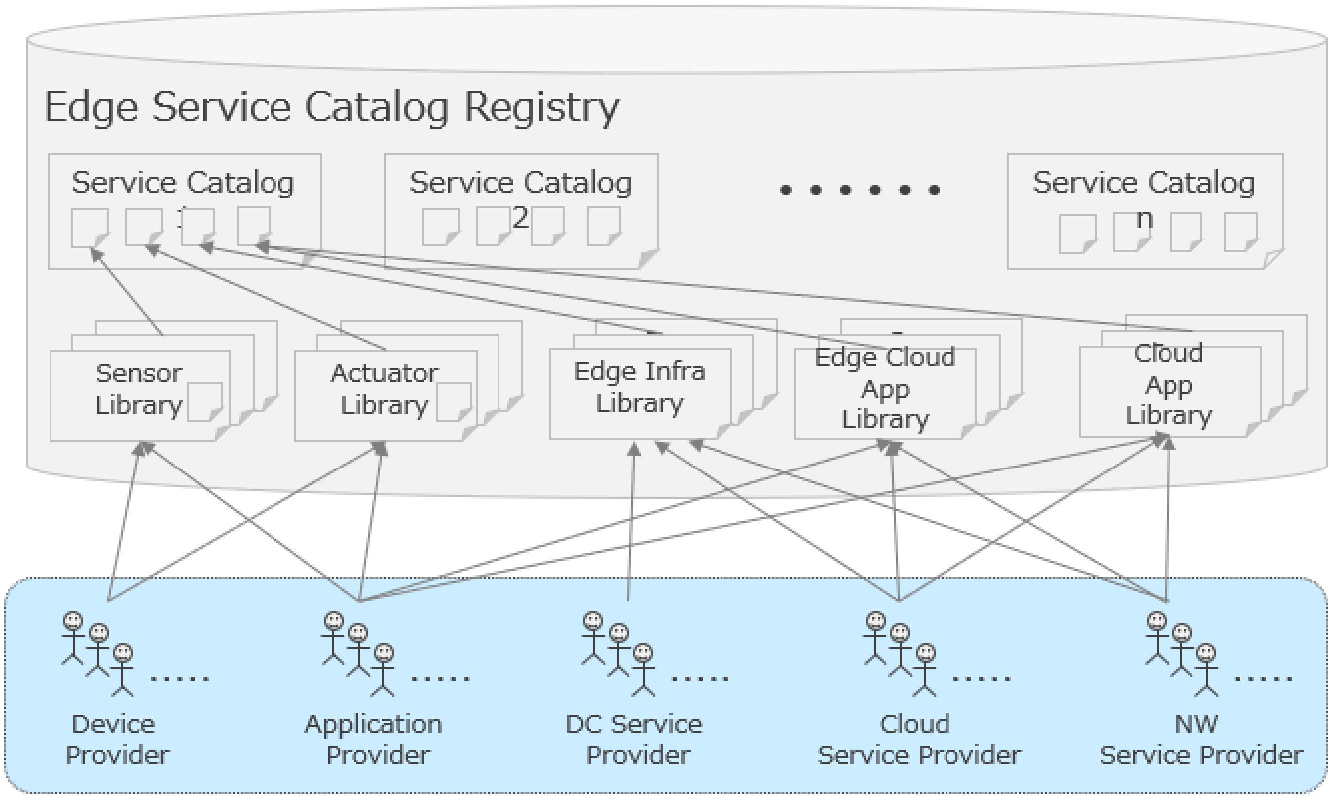
Blueprint Overview/Introduction

Designing and building edge services is made more difficult by the need to combine many different devices, infrastructure resources, infrastructure services and applications, often with their own complex dependencies and interrelationships. It is not easy for players in a particular technology or service area to acquire and maintain the necessary expertise to design and build edge services, especially as the landscape rapidly changes around them. Even so, many companies want to expand and develop their businesses by providing edge services based on their strengths in specific product and service fields.

The objective of the Edge Service Enabling Platform Blueprint is to make it possible for a community of device and resource providers and edge service designers and maintainers to cooperate to make the design and implementation of edge services easier, without requiring deeply specialized knowledge of every component that goes into the services. The intent is to provide a framework for the creation of *libraries* which encapsulate the knowledge required to deploy specific devices and resources, and *catalogs* which capture the design of edge services by combining these libraries. The intent is to make it possible for an edge service provider to select an appropriate catalog and, with a minimum of configuration, deploy it directly into their target environment.

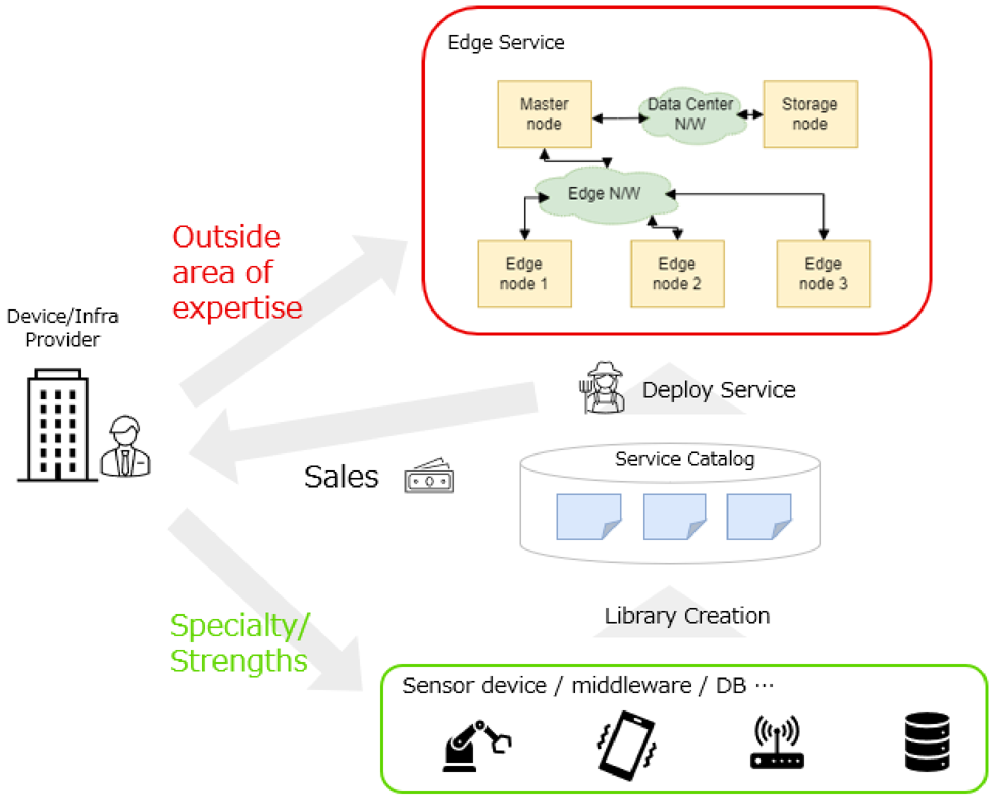


The diagram below shows how edge service catalogs encapsulate the specialized knowledge of various device, application, and service providers into a package which can be deployed in the field with minimal configuration. This benefits both the person deploying the service, by reducing the cost and effort required to design and deploy the service, and the device and infrastructure suppliers, who will see greater use of their devices and infrastructure because of the reduced friction in using them.

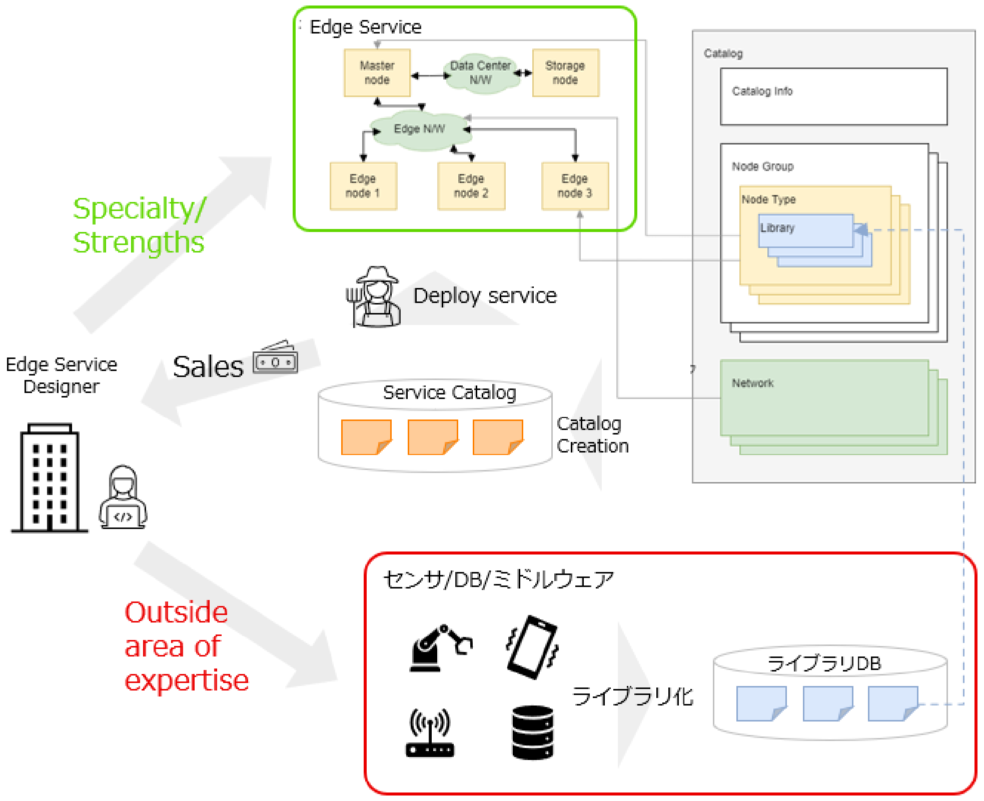


Use Cases

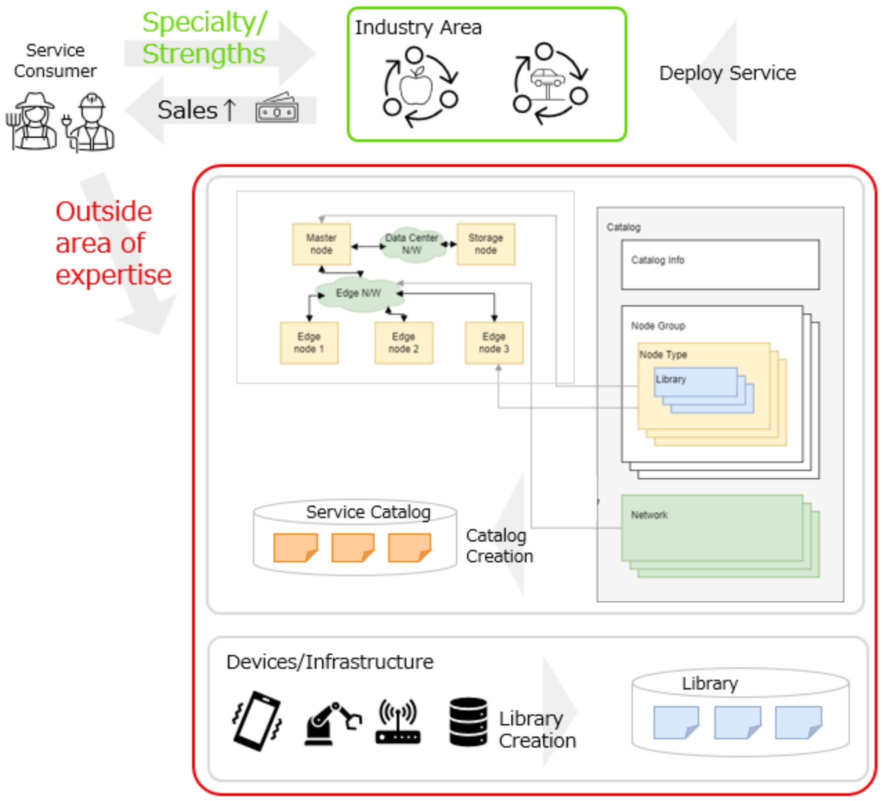
Edge Service Design & Creation:



- Device & infrastructure providers benefit from being able to provide their device or infrastructure as a library included in the service catalog, easily customized and plugged in to a variety of edge service designs, expanding the number of users for their products.



- Edge service designers can create new service designs or extend existing ones without being required to create ad-hoc solutions for each device or infrastructure component they use.



- Edge service users can choose an edge service design that best fits their needs from a variety of available designs, and easily customize it for deployment with minimal knowledge of the complexities of each component in the design.

Where on the Edge

The blueprint itself provides a service in the cloud, with connection to the edge through downloads of deployment tools and files to whatever type of target environment the service designers decide to enable. The vision is to provide the ability to deploy services arbitrarily close to the edge, possibly even including firmware or configuration of individual sensor devices. The initial release will focus on Unix-based platforms and gateways, with handling of more specialized devices to come in future releases.

Business Drivers

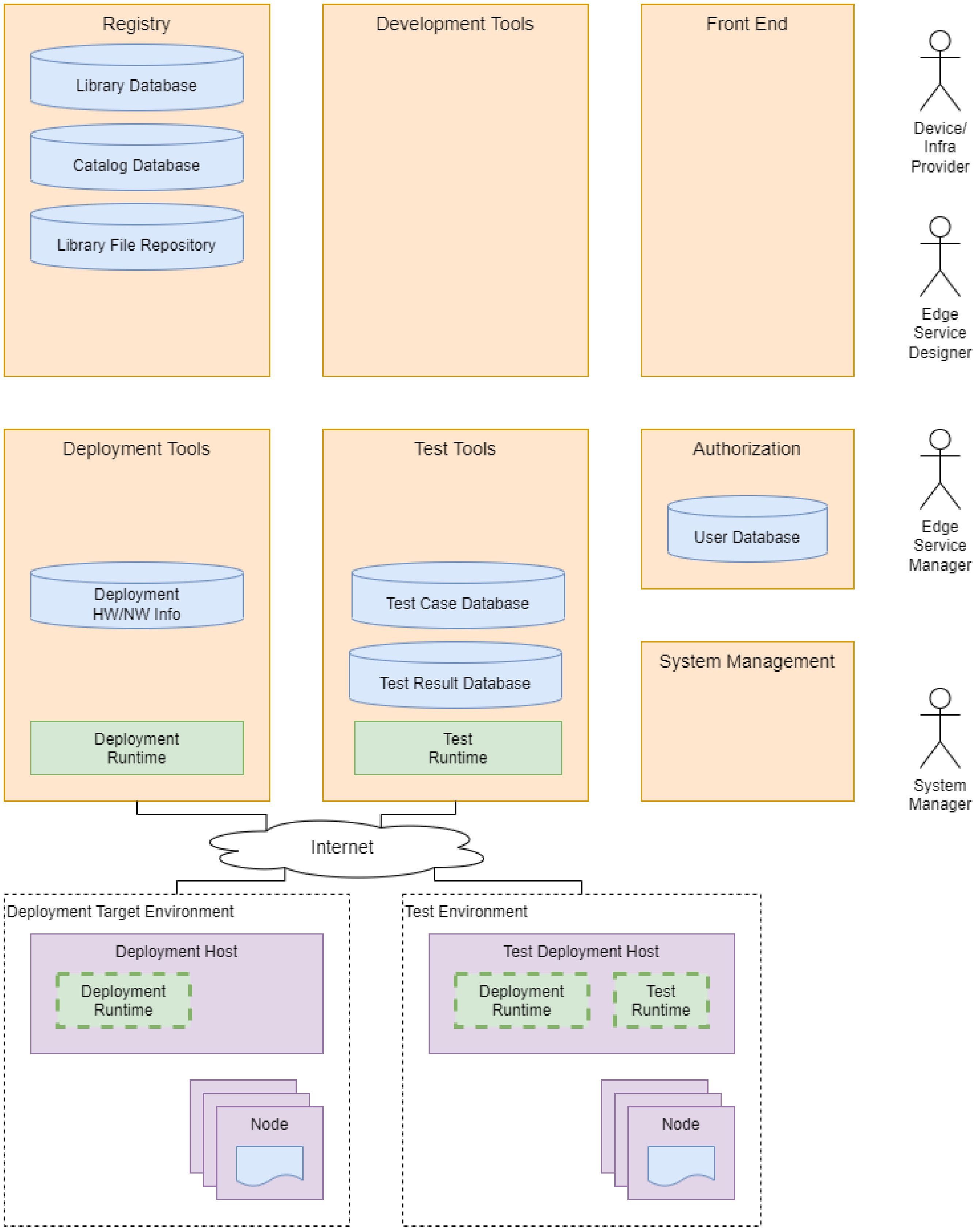
The blueprint consists of two key business driving resources: the *service catalog* and the *edge service enabling platform*.

The *service catalog* captures collective knowledge of device, application, and infrastructure experts, and in addition provides an abstract view of the various edge service components (technologies/resources/services) and the relationships between them. The service catalog enables edge service providers to focus on their strengths and easily design and develop a variety of edge services without having to go into the details of components they are unfamiliar with.

The *edge service enabling platform* provides lifecycle management for the service catalog and builds the edge services described within the service catalog, using services and resources from external infrastructure providers.

Overall Architecture

The diagram below shows the overall architecture of this blueprint.



The three main types of actor are the device/infra provider, the edge service designer, and the edge service manager. The system manager is a fourth type of actor representing the manager of the Edge Service Enabling Platform itself.

Each actor engages in the following activities using the Edge Service Enabling Platform:

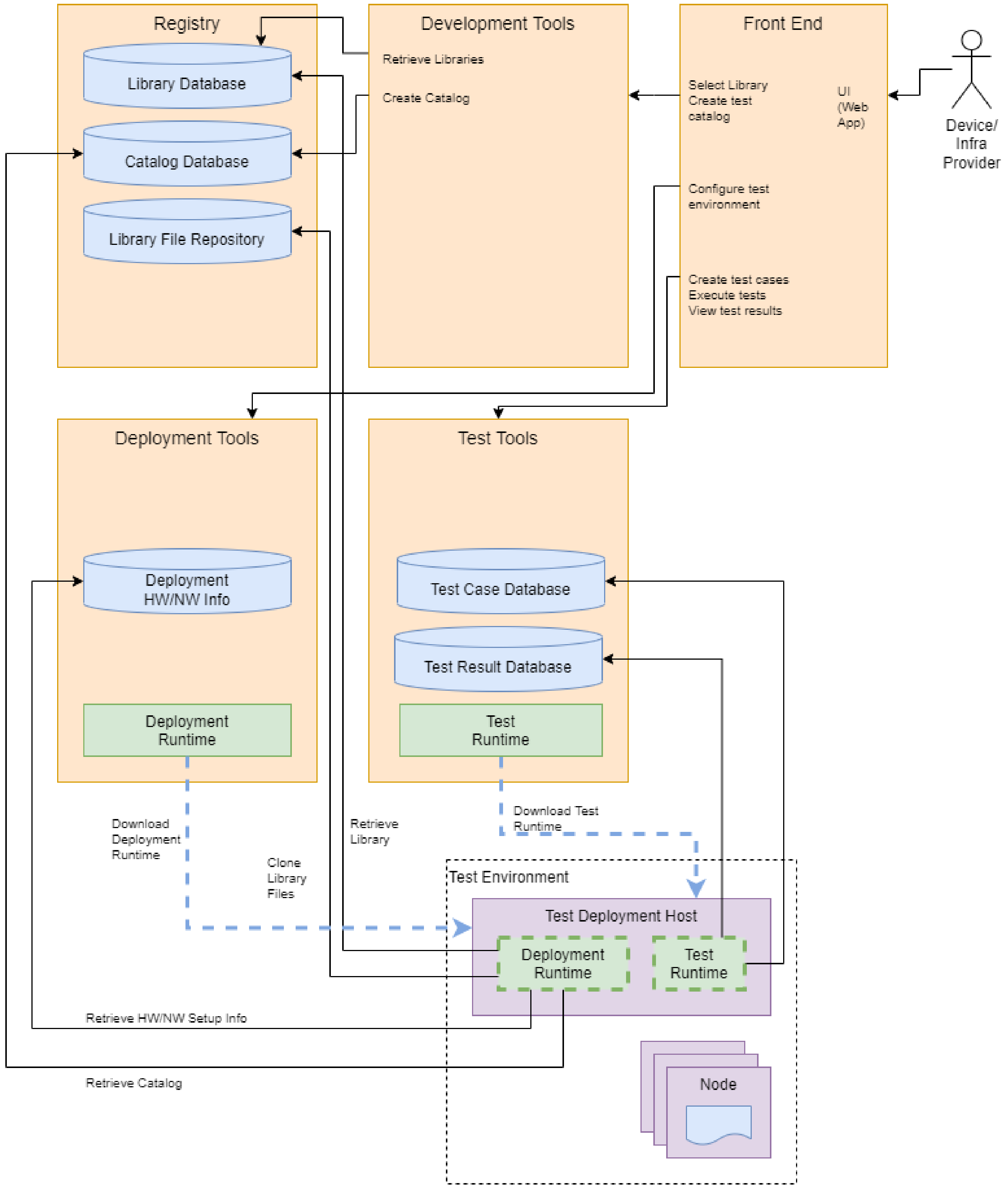
- Device/Infra Providers create and register *libraries,* which are stored in the registry component's library database and library file repository. A *librar* contains the files necessary to deploy support for a device (e.g. a temperature sensor or a camera) or piece of infrastructure (e.g. a distributed storage system, messaging system, backend database, or computing platform) in an edge service, along with the data needed to connect the library with other libraries in a complete edge service design to form an edge service *catalog*. Aside from creating and registering libraries, device and infra providers will also use test environments and the platform's test tools to verify their libraries.
- Edge Service Designers create and register edge service *catalogs*. These are stored in the registry component's catalog database and refer to libraries in the library database. A *catalog* defines a configuration of nodes and libraries deployed on those nodes, connected to each other via various styles of API (e.g. REST messaging over IP networks, Unix file sockets, device driver ioctl, or direct calls, among many other possibilities), in order to provide a service. Edge service designers will also use test environments and the platform's test tools to verify their catalogs.
- Edge Service Managers use *catalogs* they choose from the platform and the platform's deployment tools to define and deploy instances of an edge service in their target environments. They choose a catalog for the service they wish to implement, and provide the target environment for deployment. They configure the specifics of their environment (network addresses, specific hardware configuration, security keys and so on) and use the platform's deployment tools, especially the deployment runtime components, to roll out the service.
- The System Manager maintains the platform itself, configuring the underlying system, installing and updating software, making backups, and maintaining the user database.

It is possible for one organization or even one individual to perform any number of these roles, or even all of them.

To further illustrate the proposed architecture, the sections below describe the high level interactions between the platform's components when performing the three major tasks of verifying libraries and catalogs, and deploying a catalog.

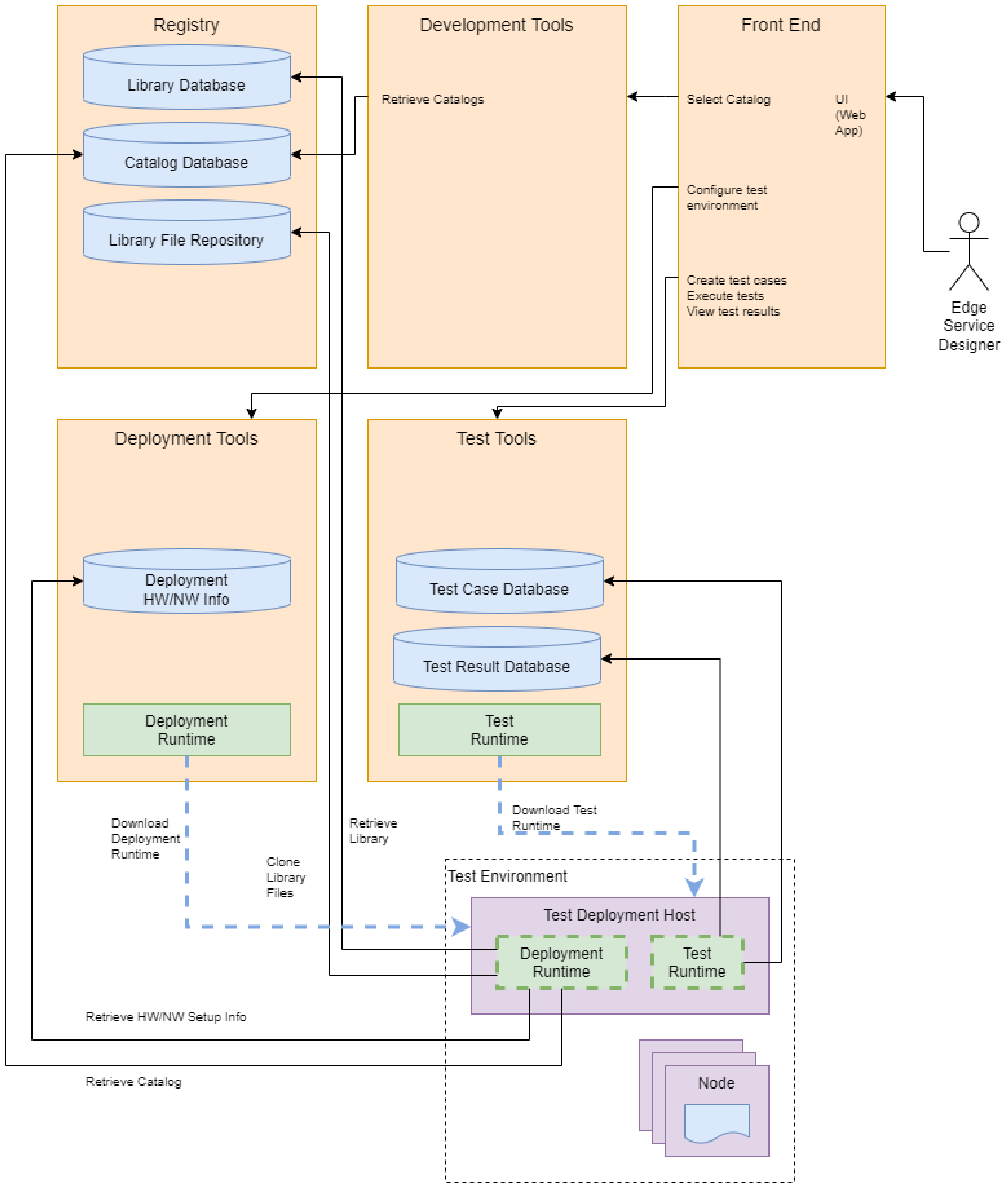
Verifying A Library

The diagram below shows the high level interactions when verifying a library using the platform. In order to verify the library operates properly, a basic catalog, consisting of the library and possibly other libraries it depends on, is defined. A set of one or more deployment configurations is also defined and stored in the deployment tools database of hardware and network information. The catalog and deployment configurations are deployed into the test environment. (The test environment could be provided by the library's developer, or it could be a shared resource.) The test runtime is downloaded from the test tools and used to run a series of test cases, which are retrieved from the test case database. The results of these tests are then stored in the test result database.



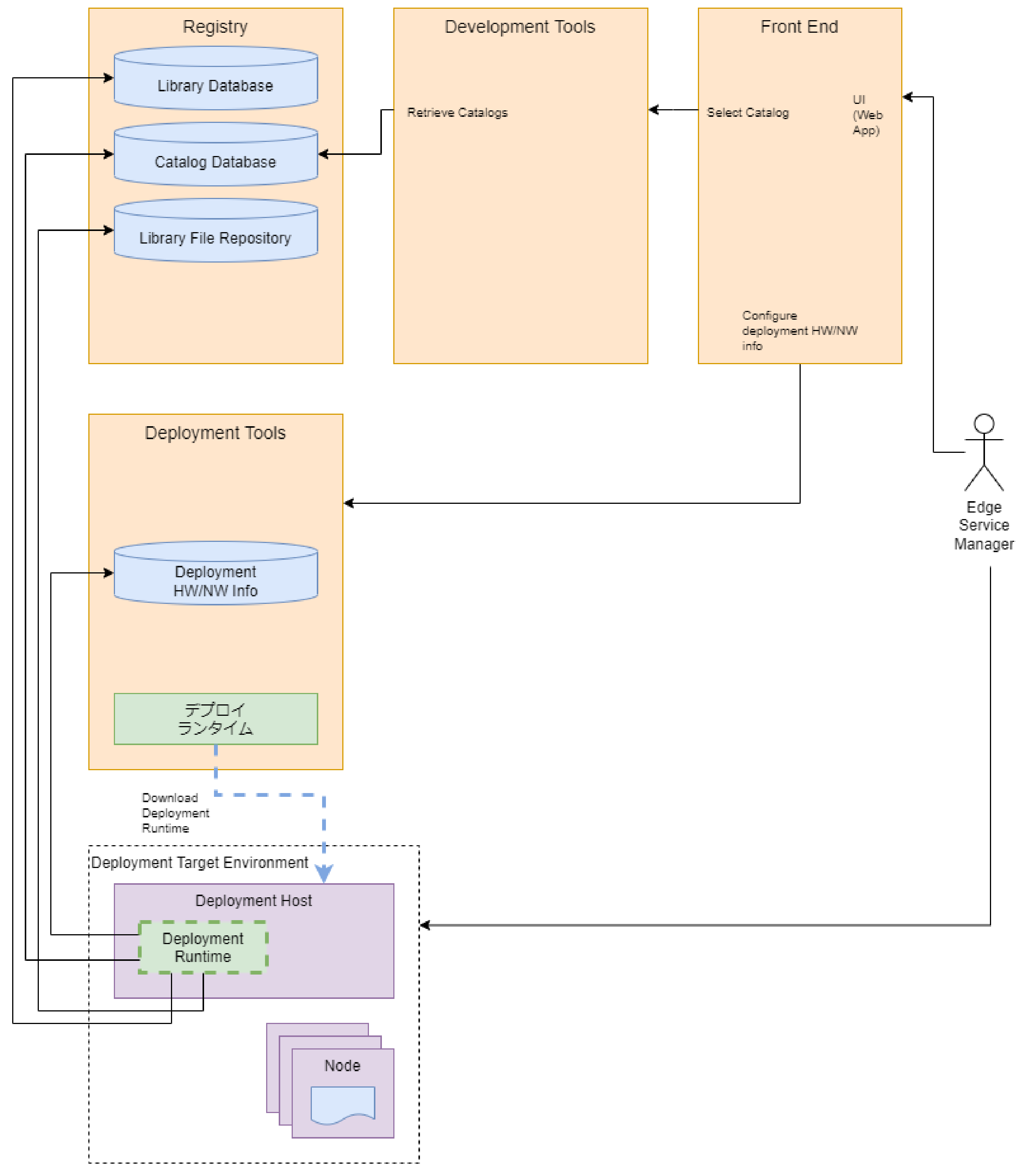
Verifying A Catalog

Verifying a catalog is very similar to verifying a library, except that the test cases focus on the operation and interaction of the libraries used by the catalog.



Deploying A Service

An edge service manager can deploy a service by selecting a catalog from the registry's catalog database, and configuring the deployment-specific information in the deployment tools' deployment hardware and network information. With that step complete the edge service manager downloads the deployment runtime to a suitable host in their target environment and the runtime retrieves the data for the required libraries, hardware and network configuration from the platform, running the deployment scripts for each library on the nodes where it is required.



Platform Architecture

The Edge Service Enabling Platform itself is intended to be implemented on a cloud platform. (The test environments and deployment target environments, of course, can have a wide variety of configurations.) The table below summarizes the platform requirements for each component in the blueprint.

	Computing	Storage	Networking
Front End	Consumer-grade x86 server or equivalent VM (> 2 cores)	Minimal storage for web application and logs (<100GB)	Public facing IP Load balancing (if user numbers require it)

			Private back-end IP interfaces to other components
Development Tools	Consumer-grade x86 server or equivalent VM (> 2 cores)	Application and log storage (< 100GB)	Back-end service IP
Registry	Managed database service or equivalent (DB sharding if loads require it)	Persistent library and catalog database storage (> 128GB) Library file repository (> 256GB)	Back-end service IP Load balancing (if loads require it)
Deployment Tools	Consumer-grade x86 server or equivalent VM (> 2 cores)	Application and log storage (< 100GB) Deployment HW/NW Info database storage (> 128GB)	Back-end service IP Public internet access for runtime push
Test Tools	Consumer-grade x86 server or equivalent VM (> 2 cores)	Application and log storage (< 100GB) Test case database storage (> 128 GB) Test result storage (> 256GB)	Back-end service IP Public internet access for runtime push
Authorization		User database storage (< 128GB)	Back-end service IP
System Management	Consumer-grade x86 server or equivalent VM (> 2 cores)	Storage for auditing logs, installation files, etc. (<500GB)	Private back-end IP interfaces for component access

Software Platform Architecture

The software components used by the blueprint are listed below. (TBD: Subject to revision and expansion as development proceeds.)

- OS: Ubuntu 20.04 or 22.04
- Services will be coded in Golang current stable release (1.18.3)
- Deployment runtime uses Ansible current stable release (5.8.0)
- Test runtime uses Robot Framework current stable release (5.0), running on Python 3.8.10
- Database components run on MongoDB Atlas instances provided by Google Cloud Platform
- Library file repositories run on Cloud Source Repository instances provided by Google Cloud Platform

APIs

No additional APIs are defined by this blueprint. (TBD: External APIs may be defined in the future.)

Hardware and Software Management

Hardware provisioning and software management is described in the installation guide.

Licensing

- Scripts and source code are licensed under the Apache 2.0 license.