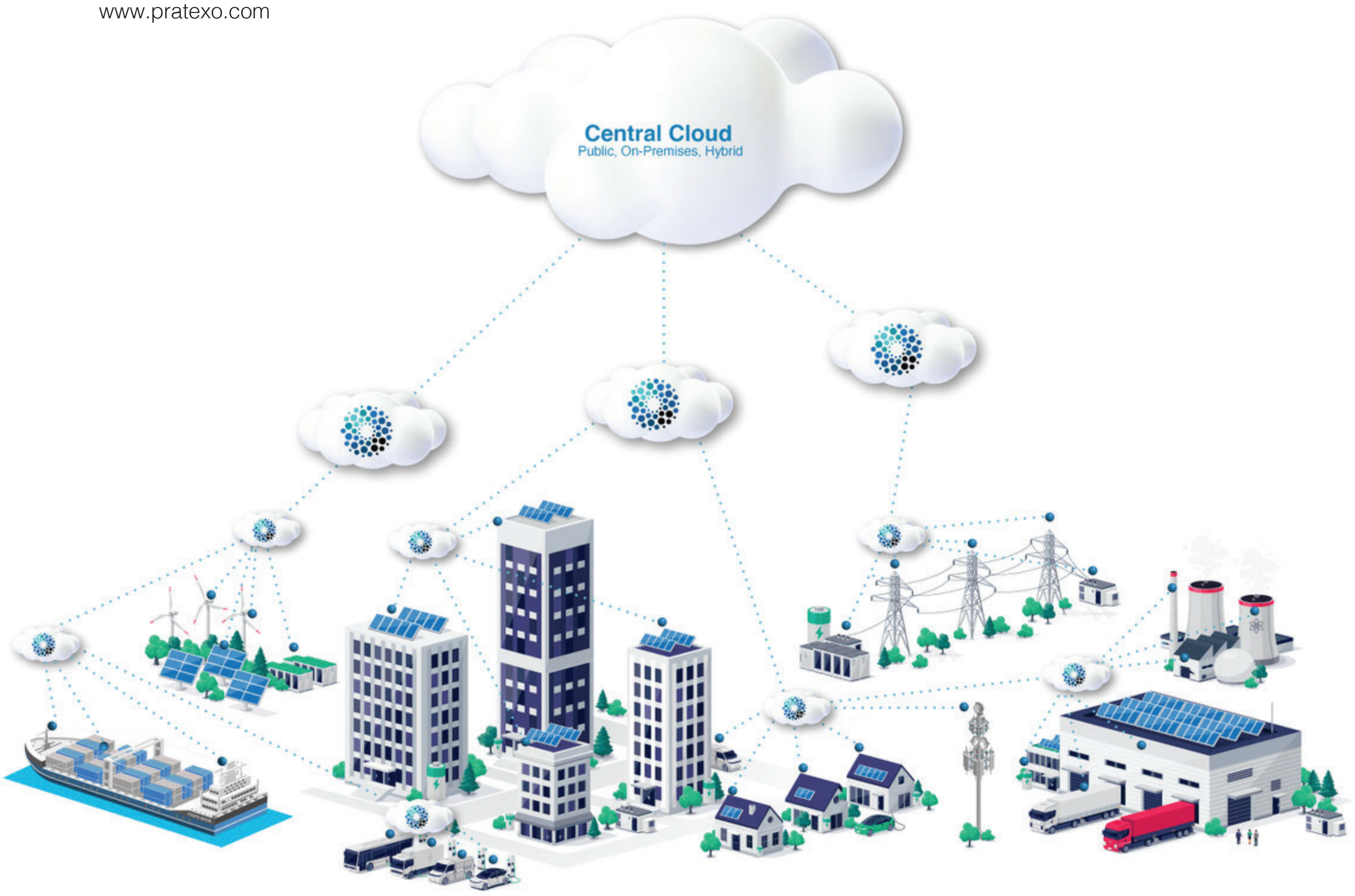
White Paper
Architecting Edge Solutions Using Pratexo

December, 2023

Petter Graff, CTO
Pratexo
petter.graff@pratexo.com
www.pratexo.com



PRATEXO

PRATEXO

White Paper
Table of Contents

Architecture Edge Solutions Using Pratexo

01 Overview
Architecting Edge Solutions Using Pratexo White Paper

At Pratexo, we specialize in crafting cutting-edge software solutions, with a keen focus on the burgeoning field of edge computing. This white paper delves into this intricate world, a domain replete with unique challenges and opportunities. We begin by exploring the multifaceted advantages and potential drawbacks of edge computing, such as its impact on data processing speeds and network reliability.

Following this, we will unveil some of the most effective architectural patterns we've identified and employed across various industries, tailored specifically for edge computing environments. These patterns not only address common industry needs but also highlight our innovative approach to problem-solving.

A critical aspect of our discussion centers on the design and development of software components. Here, we emphasize the importance of mobility, ensuring that these components are not only robust but also flexible and adaptable to the dynamic nature of edge and hybrid edge-cloud environments.

Finally, we will showcase how Pratexo's expertise simplifies the journey towards achieving optimal edge architectures. This paper aims not just to inform but to equip you with practical insights and strategies, paving the way for more efficient and effective edge computing solutions in your own organizational contexts.

02 The Advantages of Edge Computing

Edge computing offers a multitude of benefits, particularly suited to modern technological needs. Here are the key advantages:

- **Reduced latency:** Processing data near its source significantly cuts down latency, a crucial factor in use cases like industrial automation and autonomous vehicles.
- **Bandwidth savings:** Local data processing means only essential information is sent to central servers, conserving bandwidth - a vital benefit in bandwidth-limited or costly areas.
- **Improved privacy and security:** Analyzing sensitive data locally minimizes the risk of network transmission leaks, enhancing data privacy and security.
- **Reliability and resilience:** Decentralizing processing reduces vulnerability to single points of failure, ensuring system reliability even when central servers or networks are compromised.
- **Scalability:** As device numbers increase, it's more feasible to add edge computing resources than to continuously expand central data centers.
- **Efficient use of resources:** Local processing lightens the load on central servers and networks, optimizing resource utilization.
- **Support for remote locations:** In areas with poor central data center connectivity, edge computing offers essential local processing and storage.
- **Real-time analytics and insights:** Industries like manufacturing, retail, and healthcare benefit from immediate data analysis, leading to faster decision-making and operational improvements.
- **Lower costs:** Despite the initial investment, edge computing can reduce long-term expenses by cutting down on data transmission and central infrastructure costs.
- **Compliance and data sovereignty:** For legal compliance requiring data to remain within specific geographic boundaries, local processing is key
- **Customization and flexibility:** Edge computing solutions can be tailored to meet the diverse needs of various applications and locations.

These advantages collectively make edge computing an invaluable approach in our increasingly connected world, offering tailored, efficient, and secure computing solutions across various sectors.

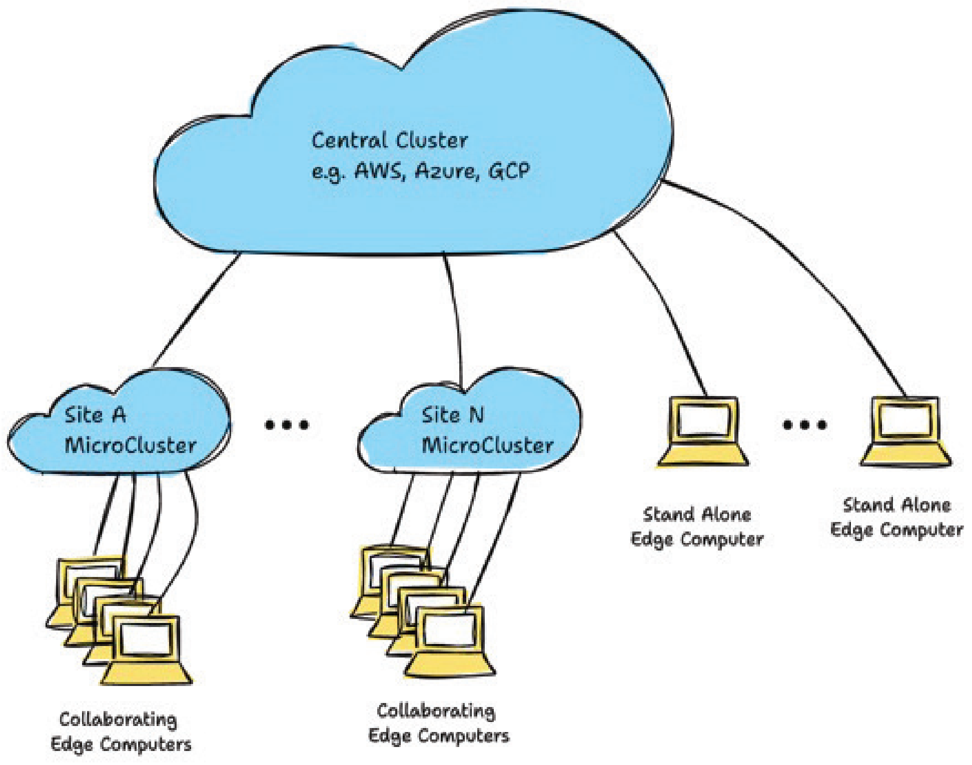
03 The Challenges

While edge computing brings numerous benefits, it also presents unique challenges, especially when compared to traditional data center environments:

- **Network connectivity issues:** Edge computing often faces poor network connectivity, which can impact data transmission and processing efficiency.
- **Availability and performance constraints:** Ensuring high availability and optimal performance in edge environments can be more challenging due to limited resources and varying conditions.
- **Environmental factors:** Devices in edge computing environments may need to withstand harsh conditions, including high temperatures and vibrations, posing significant design and operational challenges.
- **Management and maintenance complexities:** Remote locations of edge computing units can complicate management and maintenance, especially in areas lacking technical expertise.
- **Security risks:** While edge computing can enhance security in some aspects, the physical accessibility of edge devices can pose additional security risks, necessitating robust protective measures.
- **Costly scaling:** Scaling edge computing resources can be expensive, particularly when it involves physical installations in remote or difficult-to-access locations.
- **Remote installation requirements:** Setting up edge computing infrastructure often requires remote installation, which can be challenging in terms of logistics, planning, and execution.

These challenges require careful consideration and strategic planning to ensure that the deployment of edge computing solutions is effective, secure, and sustainable. Addressing these issues head-on is crucial for leveraging the full potential of edge computing technologies.

04 A Typical Hardware Configuration



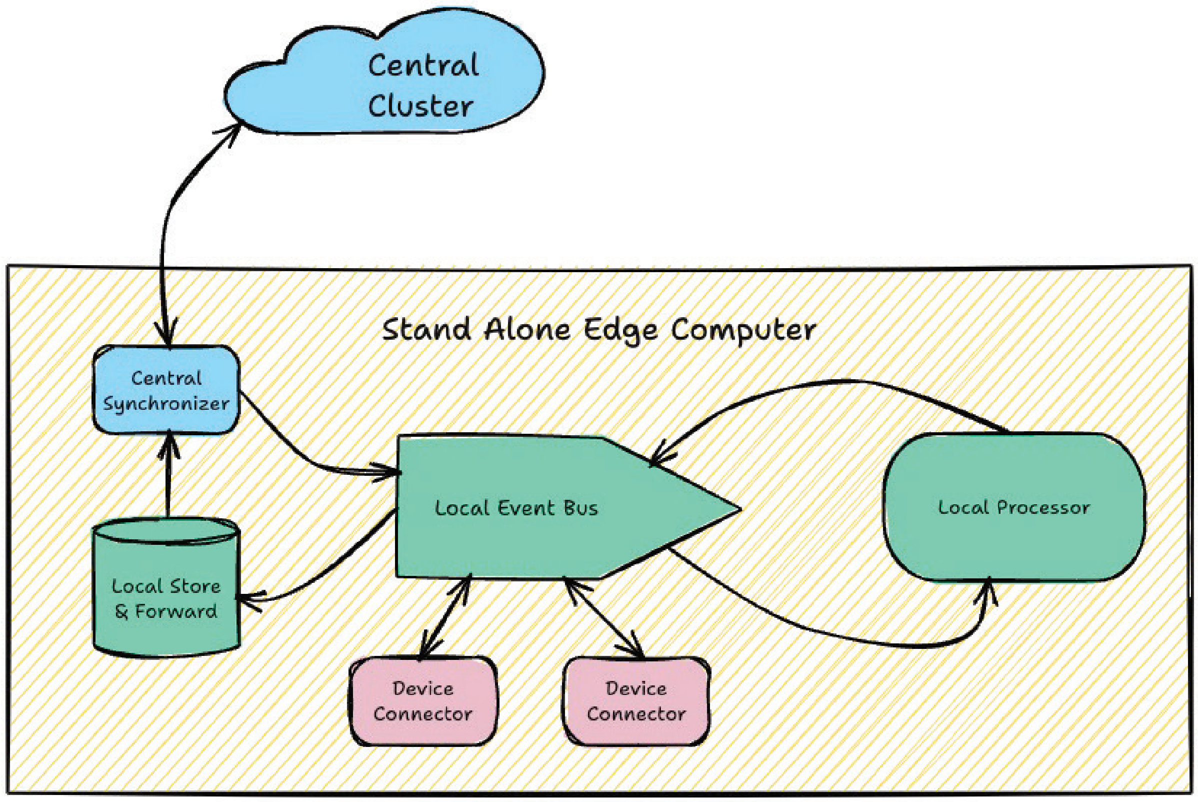
In designing edge computing solutions for our clients, we leverage a diverse array of hardware configurations to best meet their specific needs. The flexibility in our approach allows for tailored solutions that vary significantly across different solution domains.

- **Central Cloud Integration:** Often, we utilize central cloud services (such as Azure, AWS, GCP) to aggregate and analyze the final outputs from edge computing processes. This setup is ideal for scenarios requiring extensive data processing and storage capabilities.
- **Standalone Edge Computers:** In some cases, standalone edge computers are deployed to process local data. These units then transmit the processed information directly to the central cloud. This configuration is suitable for locations with specific, localized computing needs.
- **Clustered Edge Computing:** For more complex requirements, we may cluster a set of edge computers, enabling them to collaborate via platforms like Kubernetes or Swarm compute. Pratexo calls this a 'micro cloud.' Micro clouds are beneficial for tasks that require higher computing power and redundancy.
- **Hybrid Configurations:** We often see that a combination of clusters and standalone edge computers provides the most efficient solution. The optimal configuration varies, depending on the specific demands and constraints of each project.

05 An Edge Computer's Local Responsibility

An edge computer is a compute node that we place close to the devices that we control (e.g., sensors and actuators). It provides a local compute option that processes the incoming information. Part of its responsibility may be to control various actuators as well.

In most solutions we design, we have (at least) the following software components running on the edge computer.



In the majority of our designs, the following key software components are operational on the edge computer:

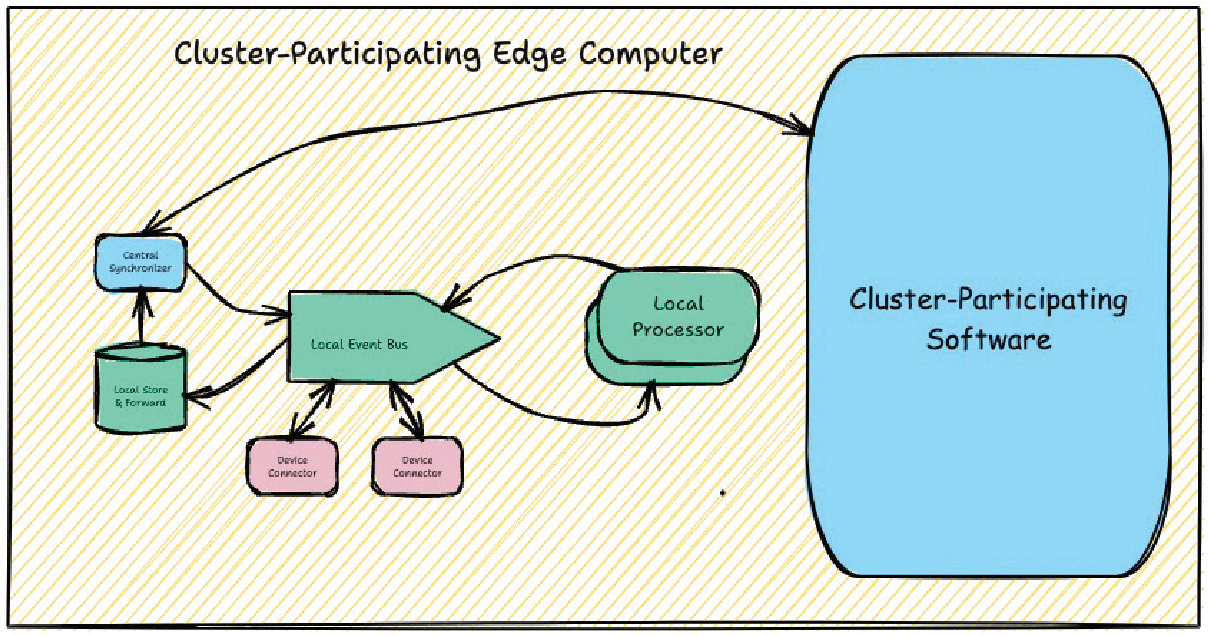
- **Device Connector:** This component interfaces with sensors or actuators, managing local solution behavior. Examples include:
    - Connecting to a Programmable Logic Controller (PLC) using the ModBus protocol.
    - Interfacing with cameras or microphones.
    - Linking to thermostats.

  In Pratexo's framework, the device connector not only gathers data but also handles control requests, interfacing with a local event bus.
- **Local Event Bus:** Acting as the communication backbone, the local event bus allows software components to publish and subscribe to information. A typical tool used here is MQTT brokers, facilitating efficient data exchange among components.
- **Local Store & Forward:** Given the often unreliable network connections at the edge, this component temporarily stores data destined for central processing. It forwards the accumulated data as soon as a stable network connection is available.
- **Local Processor:** This processor interacts with the local event bus, responding to and processing the information it receives. It can also contribute data to the bus. Common tasks include forward-chaining rules, data aggregation, and more. The choice of implementation depends on factors like resource availability and the need for unique configurations at each edge site. Solutions range from Node-Red and TensorFlow-based AI/ML applications to simple Docker containers.
- **Central Synchronizer:** This component is responsible for two-way communication between the edge and the central system. It sends data to a central cluster and disseminates commands to the edge, ensuring consistent and coordinated operation.

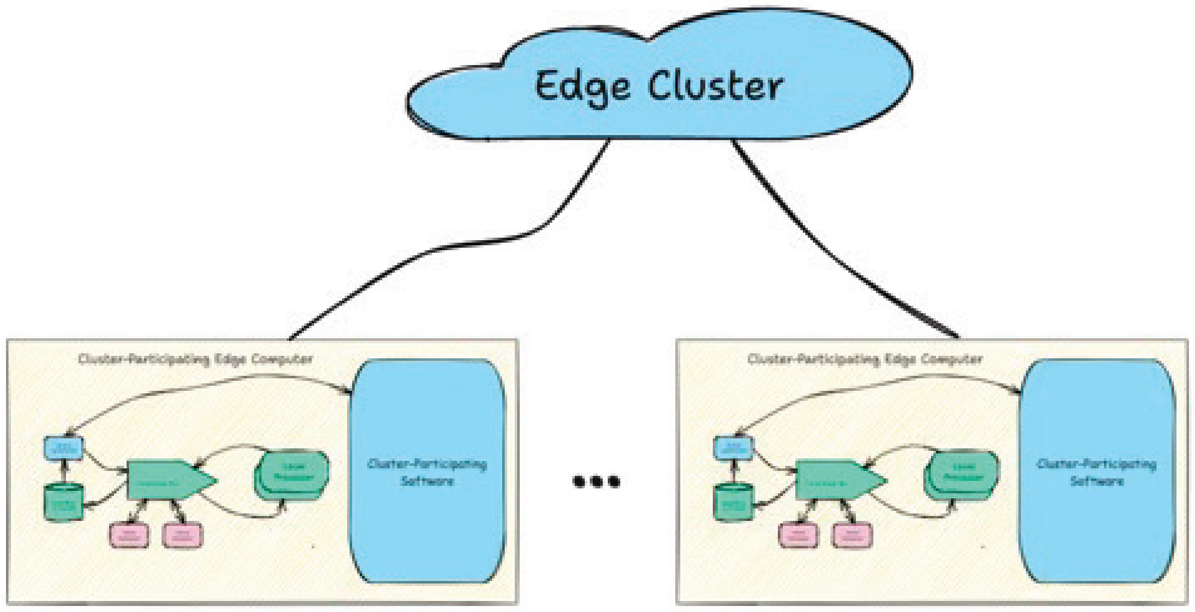
Each of these components plays a vital role in the efficient functioning of an edge computing setup, tailored to handle the unique demands of different environments and applications.

06 Clustered Edge Computing

In our edge computing solutions, clustering edge computers significantly enhances their capabilities. This approach involves configuring edge computers to work in unison, effectively creating a distributed data center.



We commonly utilize Kubernetes for clustering, often opting for specific distributions like MicroK8s, K3s, or K0s, tailored to our use cases. These distributions are less resource-intensive, making them suitable for edge environments. By clustering, each edge node contributes a portion of its resources to the collective, allowing us to deploy sophisticated software typically reserved for larger data centers.


Edge Cluster
Cluster-Participating Edge Computer
Local Processor
Cluster-Participating Software
Cluster-Participating Edge Computer
Local Processor
Cluster-Participating Software

Consider a power company with approximately 2,000 substations. By installing edge computers at these substations, each capable of local data processing and storage, we create a formidable network. Assuming each node contributes 3 CPUs, 12GB of memory, and 1.75TB of storage, the collective power equates to 6,000 Intel CPUs, 24TB of distributed memory, and 3.5PB of distributed storage. This setup not only ensures high availability and reliability but also allows for complex data processing tasks without the need to transfer data to a central cloud.

With Kubernetes, deploying data-center-grade software like Cassandra, Hadoop, MongoDB, CouchDB, RabbitMQ, Kafka, etc., becomes feasible at the edge. The system's capability scales linearly with each additional edge node, enhancing the overall processing power.

In some scenarios, simple solutions like GlusterFS for data backup across nodes are sufficient. In more dynamic environments, where node collaboration changes frequently, we might employ swarm technologies like Zenoh to facilitate flexible and efficient inter-node communication.

Clustered edge computing thus opens up new possibilities, allowing edge nodes to handle more complex tasks and collaborate in ways previously limited to traditional data centers. This approach not only maximizes the utility of each edge node but also significantly expands the overall computational and storage capabilities of the network.

Clustered edge computing thus opens up new possibilities, allowing edge nodes to handle more complex tasks and collaborate in ways previously limited to traditional data centers. This approach not only maximizes the utility of each edge node but also significantly expands the overall computational and storage capabilities of the network.

07 The Role of the Central Cloud in Edge Computing

While edge computing brings processing closer to data sources, the central cloud remains a pivotal component in our comprehensive solutions. Central cloud providers offer a range of services that are integral to maximizing the efficiency and capabilities of both standalone edge computers and edge clusters.

- **Complementary Services:** These services include scalable storage options and advanced compute capabilities, which are essential for handling tasks that are beyond the scope of edge computing. For instance, the central cloud can perform large-scale data analytics, leverage advanced AI/ML algorithms, and aggregate data from multiple edge locations for global insights.
- **Integration and Scalability:** Integrating edge computing with central cloud services allows for a more scalable and flexible architecture. This synergy enables our solutions to adapt to varying workloads and data demands, ensuring optimal performance across all levels of the network.
- **Data Management and Compliance:** In most of our solutions, we incorporate some form of centralized computing. This approach is recommended to our clients, except in cases where regulatory constraints or specific data privacy concerns necessitate keeping data out of public clouds. Understanding and adhering to these compliance requirements is a key aspect of our solution design.

For example, in a scenario where edge devices collect real-time data across multiple retail locations, the central cloud can aggregate this data to provide comprehensive analytics and insights, enabling strategic decision-making at the corporate level.

The central cloud plays a complementary and often essential role in edge computing architectures. It offers the scalability, advanced processing capabilities, and global data management necessary to realize the full potential of edge computing solutions, ensuring they are robust, versatile, and compliant with regulatory standards.

08 Software Mobility at Pratexo

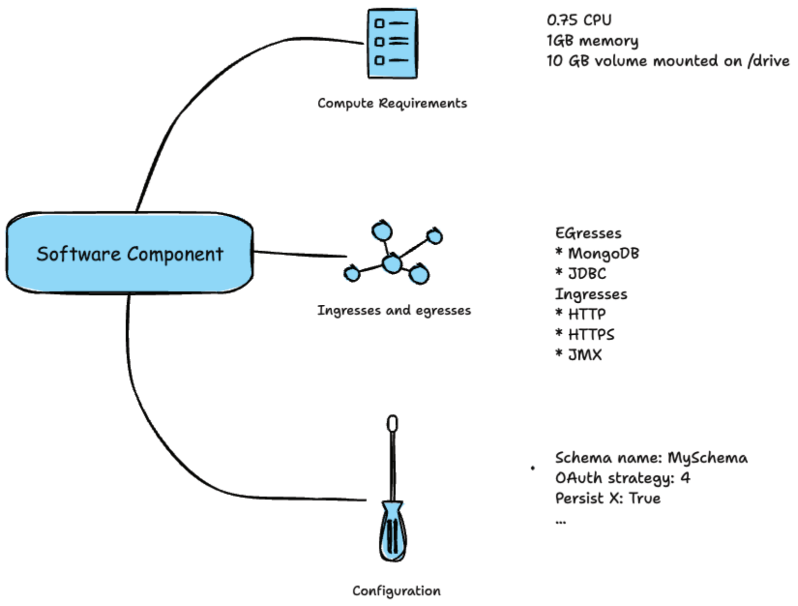
At Pratexo, we prioritize software mobility, ensuring that our software components can be seamlessly moved and deployed across various environments, from edge to on-premises private clouds to hyperscaler clouds, without extensive modifications. This flexibility allows solution providers to experiment with different deployment scenarios, understanding the trade-offs and benefits of each.

Why software mobility? Software mobility is crucial for two primary reasons:

- **Supporting Diverse Deployment Models:** It enables the deployment of solutions in a variety of environments, catering to different operational needs and scenarios.
- **Post-Optimization Flexibility:** It allows our customers to optimize how their solutions run post-deployment, without the need for code changes.

Achieving software mobility involves adhering to certain technical prerequisites:

- **Defined and Mobile Dependencies:** The software must have clearly defined dependencies, all of which must be portable across desired environments.
- **Configurability:** The software should be easily configurable in any environment where it might be deployed.

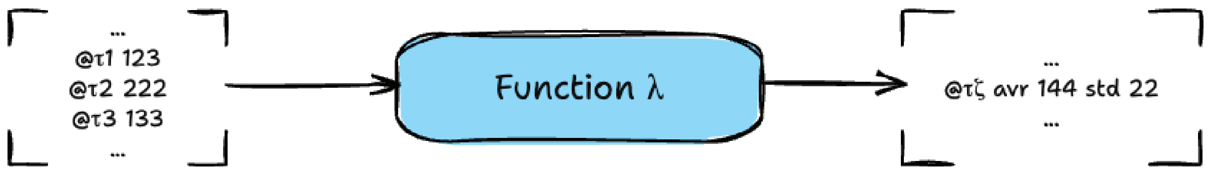


For Pratexo today, this means:

- **Containerization:** Our software is designed to run in containers (e.g., Docker containers) that are compatible across all relevant architectures. For instance, software designed for Intel CPUs must also be adaptable to ARM processors.
- **Defined Ingresses and Egresses:** All inputs and outputs, including network connections, must be explicitly defined.
- **Resource Requirements:** The software's requirements for compute and storage must be clear and adaptable to varying loads. This includes specifications for memory, CPU, and disk space.

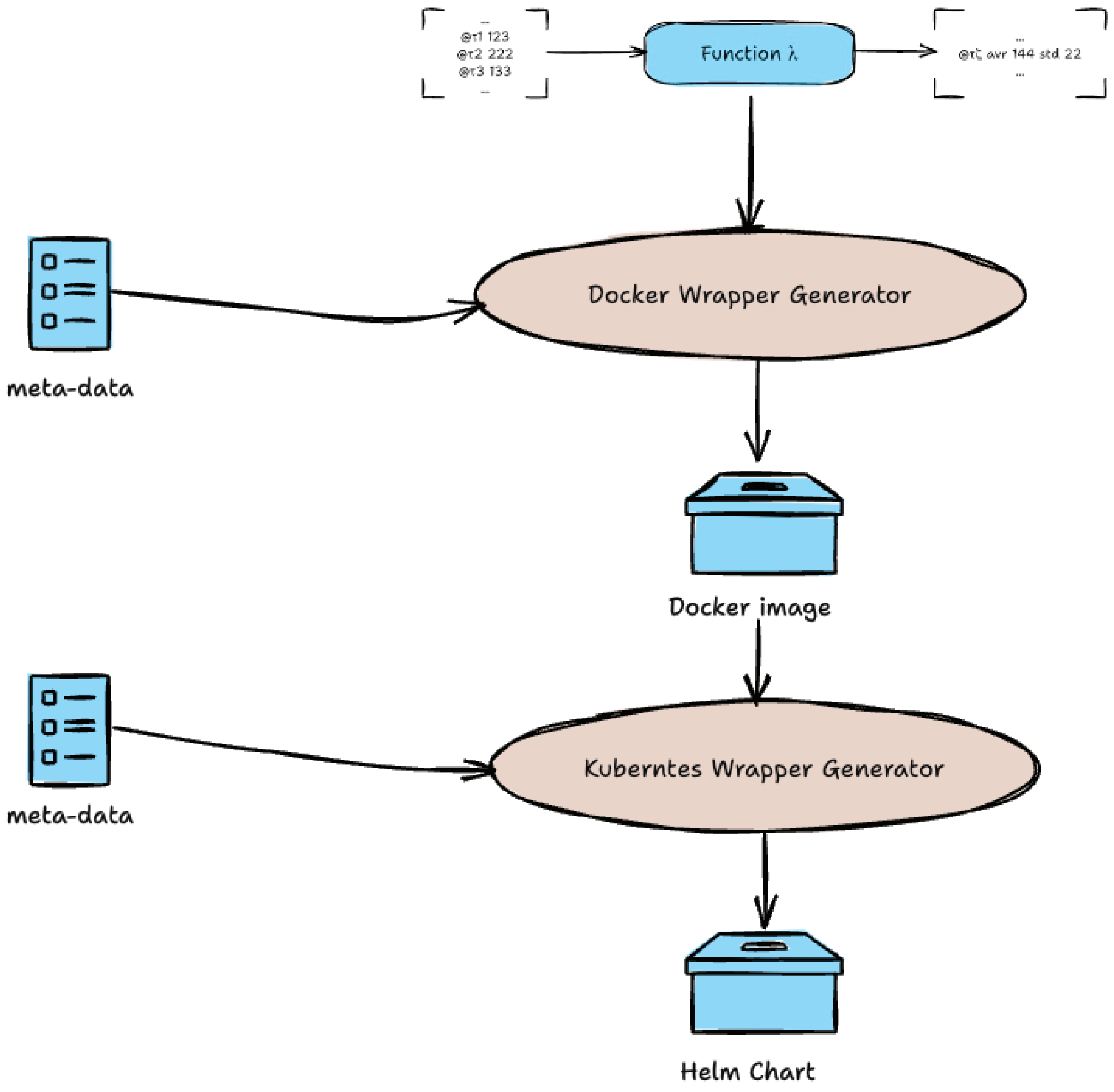
Most of the software that has to be mobile is in most cases much simpler than what's above. In most cases, the software can be thought of as a function with some input and some output and using various wrappers, we can generate the software that surrounds the function.

Say for example that we have some software component that processes sensor events and publishes the average value over some time interval.



Say we implement this function in Rust and we know the input comes in the form of MQTT messages at some topic and we want to produce the average on some other topic.

Let's say we want to be able to deploy this function on the edge running in a Docker container and also as a function in kubernetes. All the code required to make this function available in a Docker container is generatable. That is, we have to create a Docker file with two phases, one to compile the code, the other to start the code. We may have already built a wrapper in Go to receive MQTT messages and produce MQTT messages (in Pratexo's case, we have) so all we have to do is to add this launcher and point it to the function and compile it all together.



If we had to deploy it in Kubernetes, all we would have to do is to wrap the Docker container in a helm chart, also generatable.

09 Compute Where Optimal

A well-known adage in the Big Data community states, “Don’t move your petabyte to the algorithm; move your algorithm to the petabyte.” At Pratexo, our commitment to software mobility allows us to adhere to this principle with relative ease, optimizing computing efficiency and cost.

Consider a case with one of our clients who developed patient monitoring systems for home healthcare. Initially, their setup involved multiple sensors in each home, transmitting data directly to the cloud for processing. This method, while effective, incurred substantial operational costs, reaching millions of dollars monthly.

Upon analysis, we identified an underutilized resource: an Android tablet provided to each patient, primarily used for video conferencing. We explored the possibility of leveraging the tablet's processing power (originally a dual-core, now often an octa-core) to run the patient monitoring algorithm locally.

The transition was successful. By rerouting the sensor data to the tablet and processing it locally, we significantly reduced cloud dependency, resulting in considerable savings in both cloud computing and data transmission costs.

This solution was facilitated by natural data partitioning – each patient's data could be processed independently on their respective tablet. In other scenarios, where data isn't as naturally partitioned, we've sometimes deployed partial processing at the edge, synthesizing results using advanced concepts like monoids or monads.

This approach is not just about cost-saving; it's about optimizing resource utilization and enhancing system efficiency. By computing data where it's most logical and efficient, we can tailor solutions that are both economically and operationally optimal.

In conclusion, the principle of computing where optimal is central to our approach at Pratexo. It guides us to create more efficient, scalable, and cost-effective solutions, demonstrating the power and flexibility of edge computing in various scenarios.

10 Edge Orchestration from the Start

Edge orchestration refers to the process of systematically managing, coordinating, and optimizing the deployment and operation of composable computing services, applications, and resources at the edge of a network.

This concept is particularly relevant in the context of the Internet of Things (IoT), 5G networks, and distributed computing environments. It is essential to think about this from the very beginning of the project. Some of the key aspects to take into account are:

- **Resource Management:** Efficiently allocating and utilizing computational resources, storage, and networking capabilities at the edge. This involves ensuring that the right resources are available at the right time to meet the demands of applications and services.
- **Service Deployment:** Automating the deployment of applications and services to edge devices. This can involve containerization and other technologies that allow for lightweight, flexible deployment.
- **Data Processing and Analytics:** Facilitating real-time or near-real-time data processing and analytics at the edge, reducing the need to transmit large volumes of data back to a central location. This is crucial for applications requiring low latency or operating in bandwidth-constrained environments.
- **Security and Compliance:** Ensuring that data and services at the edge are secure and comply with relevant regulations and policies. This includes managing data privacy, access controls, and threat detection and response.
- **Scalability and Flexibility:** Enabling the edge infrastructure to scale up or down as needed and to support a wide range of applications and services.
- **Interoperability and Integration:** Ensuring that edge computing solutions can work seamlessly with existing systems and technologies, including cloud services and legacy infrastructure.
- **Automation and Self-Healing:** Implementing automated processes for monitoring, maintenance, and recovery to enhance reliability and reduce the need for manual intervention.

Edge orchestration is a critical component in modern distributed computing architectures, as it helps organizations leverage the full potential of edge computing, including improved response times, enhanced data privacy, reduced bandwidth costs, and the ability to operate in remote or disconnected environments.

It is essential that these aspects are brought into the implementation from the very beginning of the project. Many of the solutions we've evaluated have edge orchestration as an afterthought. It is very hard to retrofit these features into an existing architecture.

11 The Pratexo Approach

Deployable Units – Features

At Pratexo, we have a fundamental concept central to our approach: "Features." These are deployable software components, meticulously designed for easy installation across various environments. But at Pratexo, a feature is more than just a software package; it's a versatile unit of deployment that can take various forms:

- **Deployable Actions:** These could be as fundamental as a kernel patch in Linux or as routine as initializing a database.
- **Configuration Elements:** This includes setting up monitoring and analytics dashboards in tools like Grafana, Kibana, or Zabbix, or defining specific flows in Node-Red.
- **Deployment Scripts:** We utilize a range of tools for efficient deployment, including Ansible, Chef/Puppet, SaltStack, and Terraform, among others.

To construct these diverse and complex features, we've developed a suite of tools, including advanced code generators. A pivotal tool in our arsenal is the Pratexo Feature Studio, which plays a crucial role in streamlining the deployment process. This tool, which I will elaborate on in a later section, is integral to our ability to deliver tailored, efficient, and robust software solutions.

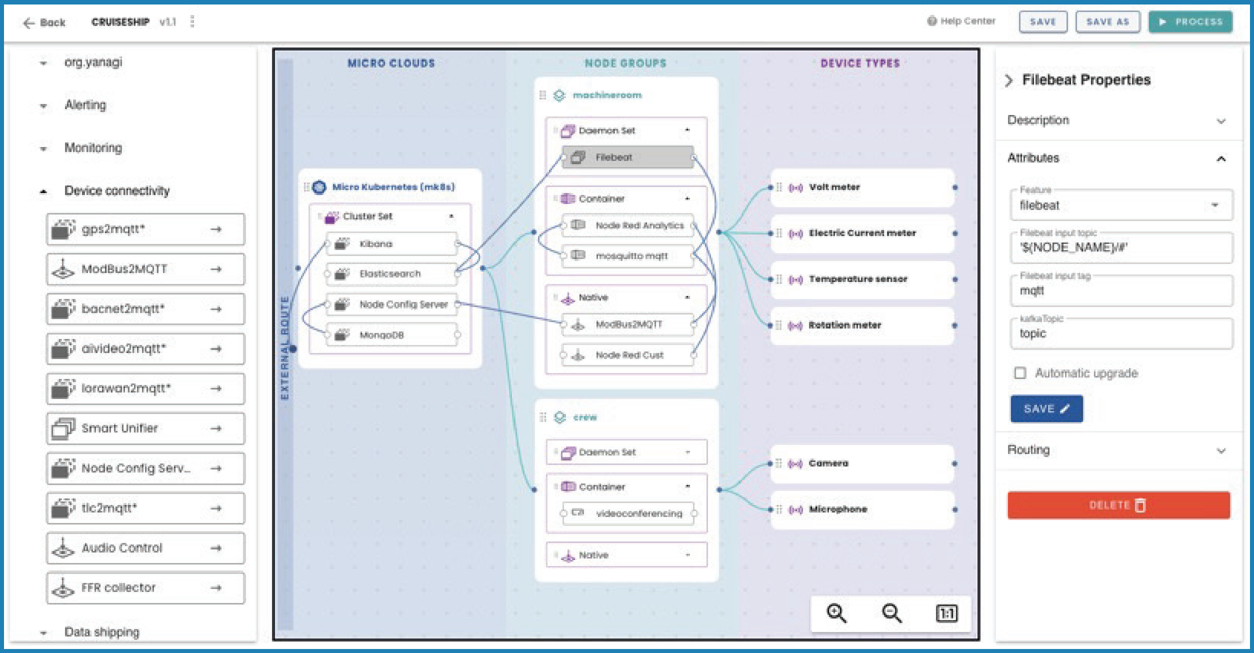
In essence, our approach to features at Pratexo is about ensuring versatility, ease of deployment, and adaptability, aligning with the diverse needs of our clients and the dynamic nature of technology environments.

Pratexo Design Studio

At Pratexo, we've developed a unique and flexible design tool to architect sophisticated systems: the Pratexo Design Studio (PDS). This tool stands out in its ability to not only construct architectures using existing software components but also identify and plan for components that might be needed in the future.

PDS operates by configuring a set of software components and defining how they interact. These components, along with their requirements and configurations (referred to as 'metadata'), are crucial in the design process.

Here is an example of a screenshot from PDS:



Filebeat Properties

Description

Attributes

Feature: filebeat

Filebeat input topic: '${NODE_NAME}/#'

Filebeat input tag: mqtt

kafkaTopic: topic

Automatic upgrade

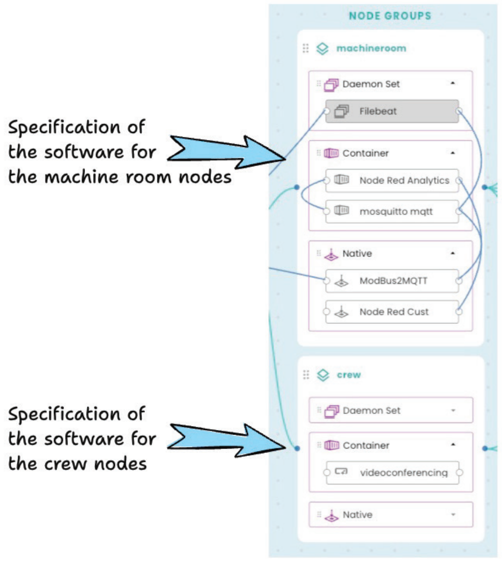
SAVE

Routing

DELETE

Consider a hypothetical scenario where we're designing a system for a ship. In this setup, we have edge nodes in the machine room for real-time analytics of engine performance and separate computing systems in crew cabins, primarily for video conferencing.

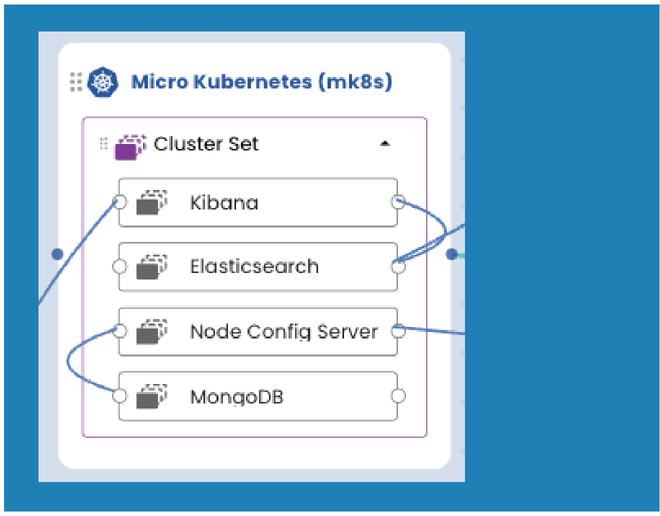
In PDS, we categorize these installations as 'Node Groups.' For instance, in the machine room Node Group, we might include components like ModBus2MQTT, Node-Red, Mosquitto, and Filebeat. Conversely, the crew cabins would have a different set of software, tailored to their specific needs.



In the machine room, we plan to install ModBus2MQTT, Node-Red, Mosquitto, Filebeat, etc. For the crew, we're only installing some video conferencing software.

Each box within a Node Group in PDS represents a deployable feature, with details on installation methods – whether in containers, natively, or via Kubernetes as daemon-sets.

Further, we specify how these edge nodes form a cluster (micro cloud) using MicroK8s, transforming the ship into a floating data center. This cluster could host software like Kibana and Elasticsearch, utilizing the spare capacity of edge nodes for data analysis and machine learning tasks.



On the left-hand side, you see a list of some of the software components or actions that we have learned how to install and that have been featurized using our Pratexo Feature Studio.

Perhaps one of the most intuitive aspects of PDS is the representation of connections between components. For example, a line between Kibana and Elasticsearch in PDS isn't just a line; it represents a complex service configuration in Kubernetes using ClusterIP. Our DevOps team creates scripts to configure these connections, simplifying the architect's job to essentially drawing these lines.

Pratexo Feature Studio

At the heart of our software development process is the Pratexo Feature Studio, a tool specifically designed to empower software providers in defining and deploying their software. This studio stands out for its seamless integration with various CI/CD pipelines, enhancing the development and deployment workflow.

One of the pivotal features of the Feature Studio is its ability to locally validate deployable software components. This ensures that each component is reliable and consistent, meeting our high standards before it becomes part of a larger system.

The primary goal of the Feature Studio is to streamline the development process. It complements our Pratexo Design Studio, allowing for an efficient transition from the creation of software components to their management and deployment within system architectures. By leveraging the Feature Studio, developers can focus more on innovation and less on the intricacies of deployment logistics.

The integration capabilities of the Feature Studio with existing CI/CD pipelines mean that software providers can easily incorporate it into their existing development processes. This integration not only enhances workflow efficiency but also ensures that the components developed are optimized for deployment through the Pratexo Design Studio.

In summary, the Pratexo Feature Studio is an essential tool in our ecosystem, simplifying and enhancing the way software components are developed, validated, and prepared for deployment in diverse and complex system architectures.

Simulating and Testing Architectures in Edge Systems

In the complex world of edge systems, thorough simulation and testing are paramount. This process ensures that systems are robust, reliable, and ready for real-world deployment. Drawing inspiration from high-level simulations, like those used by NASA for the Space Station, we at Pratexo have developed a more accessible yet equally effective approach for our clients.

While most companies cannot replicate NASA's extensive simulation setup, the principle of creating a realistic test environment remains vital. Our solution, integrated into the Pratexo Design Studio (PDS), offers a practical and cost-effective alternative.

PDS facilitates two primary types of deployments:

- **Physical Deployment:** This involves installing the software on actual physical servers and edge nodes, mirroring the intended real-world environment.
- **Simulated Deployment:** Here, the software is installed on virtual machines in the cloud, complete with device simulators for emulating inputs from field devices. This type of deployment is invaluable for testing solutions in a controlled, risk-free environment, allowing for thorough evaluation and fine-tuning before physical deployment.

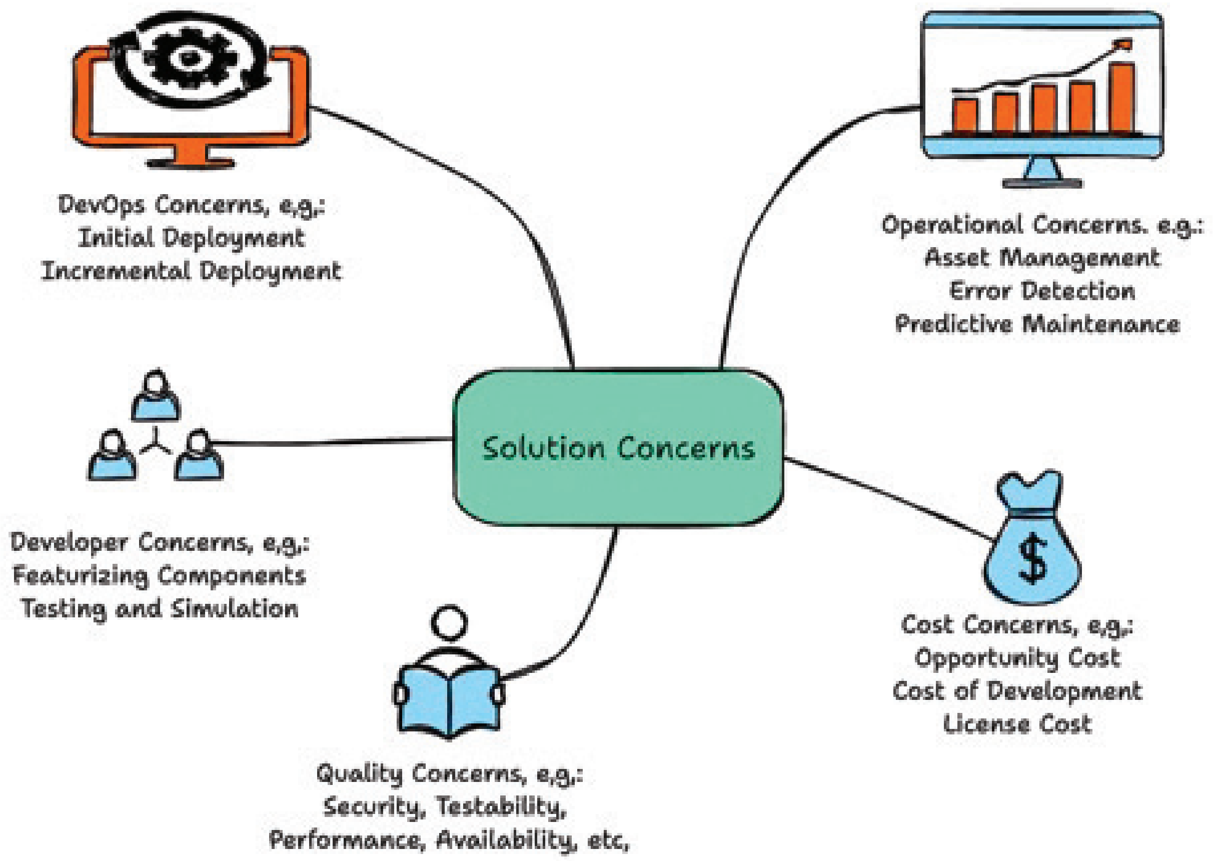
Simulated deployment offers several benefits. It significantly reduces the costs and risks associated with direct physical testing, allows for early detection of potential issues, and provides a flexible environment for testing various scenarios and configurations.

Looking ahead, we plan to introduce elements of network chaos and other challenging real-world scenarios into our simulations. This enhancement will allow for even more rigorous testing, ensuring that our solutions are not just functional but also resilient under various stress conditions and unexpected disruptions.

The ability to simulate and test architectures comprehensively is a cornerstone of our approach at Pratexo. It ensures that the solutions we deploy in the field are not only innovative but also tested against a spectrum of scenarios, guaranteeing reliability and effectiveness in real-world applications.

Benefits of Using Pratexo

When you build an edge solution, you have to balance a set of concerns.



Our solution at Pratexo offers a range of compelling benefits to our clients, designed to enhance efficiency, streamline processes, and leverage expert knowledge:

- **Enhanced Productivity:** Our ability to design and deploy architectures rapidly is unparalleled. One of our clients, a large systems integrator, remarked, "What you guys did in minutes would have taken my engineers months to put together." This level of efficiency translates into significant time savings and faster project turnaround.
- **Order in Software Portfolios:** For clients managing extensive software portfolios, Pratexo brings much-needed order and clarity. By requiring software assets to be thoroughly described and their dependencies captured, we often enhance the accessibility and reuse of existing components, streamlining the entire software ecosystem.

- **Expert Implementation:** The components and configurations in Pratexo are crafted by experts in the field. This expertise is embedded in every installation, ensuring high-quality, reliable, and effective solutions from the get-go.
- **Headstarts with Reference Architectures:** We provide a suite of reference architectures, offering architects a solid and tested starting point for their projects. These can be adapted to specific needs, ensuring a robust foundation and accelerating development timelines.
- **Mobile Components:** The Pratexo Feature Studio encourages the architect/designers to build components that can be installed everywhere making it easier to optimize the solutions.
- **Out-of-the-Box Open Source Components:** Pratexo has already "featurized" many of the commonly used open source components. Some of these components are difficult to configure to run on the edge and we've simplified the configuration to make it easy for an architect to bring them into their design.
- **Edge Orchestration from the Start:** The Pratexo Design Studio provides some of the features that are required for edge orchestration. In addition, we have featurized a set of components that allows us to monitor and control the application in the field. This includes aspects such as monitoring, autonomous installation, incremental upgrades, dashboards, and much more.

In summary, Pratexo stands not just as a tool or a platform, but as a comprehensive solution that empowers businesses to achieve more with less, leveraging expert knowledge and efficient design for optimal results.