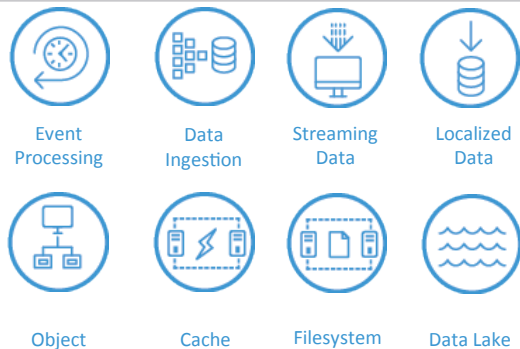


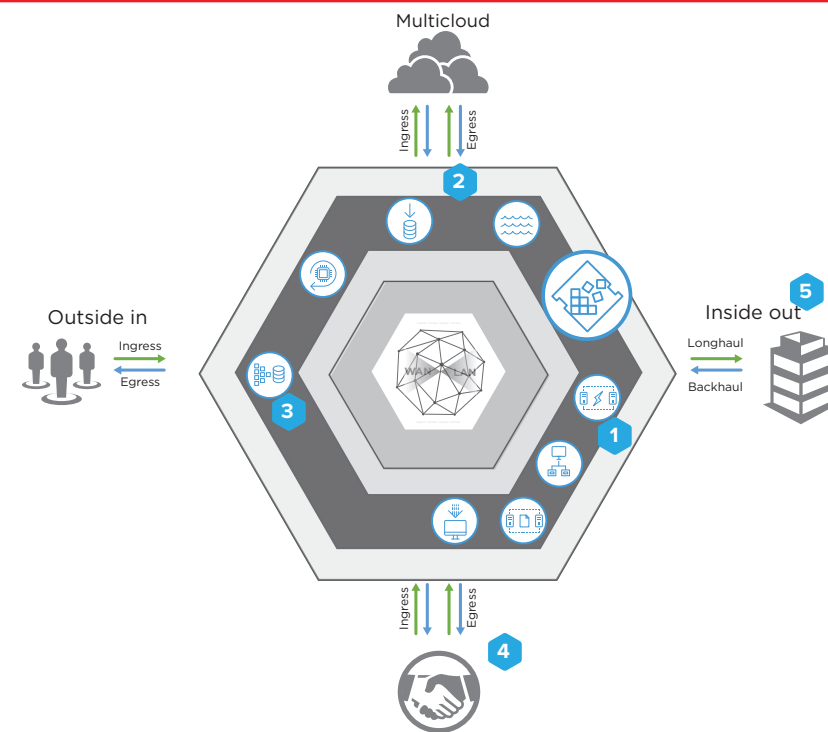
Design Principles

- Data accessibility.
- Data locality (localize data).
- Data services (virtualize data interfaces).
- Data layers (batch, speed, serve).
- Data regulation (data sovereignty).
- Data privacy, encryption, masking and anon.
- Data provenance and governance.

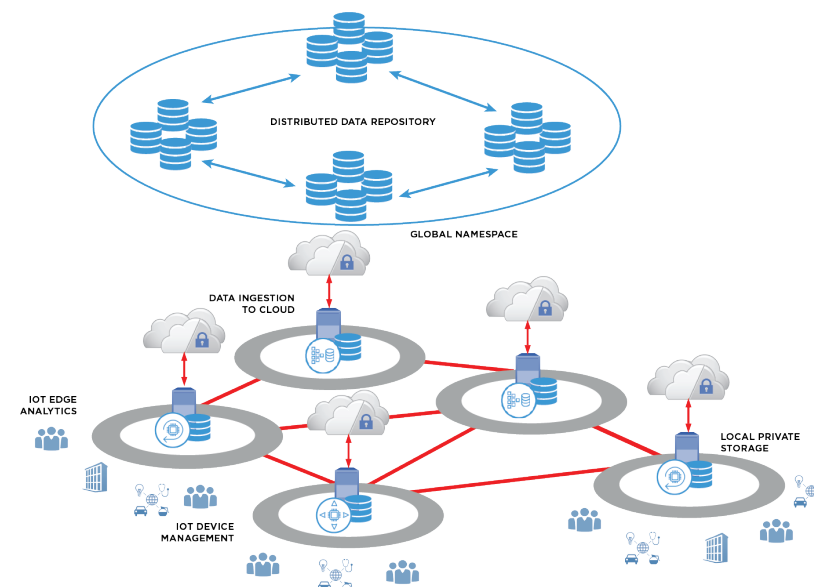
Edge Node Components



To architect for the digital edge, you need to localize some data requirements in the digital edge node, balance protection with accessibility, and govern data movement and placement. Each node is tailored for the local or shared data services at that geographic location, placing you in control of your data and performance.



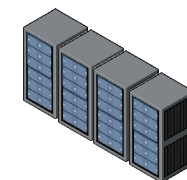
Establish a Data Fabric



Capabilities

- Secure data traffic with traceability.
- Optimized data placement for performance.
- Multicloud application workloads have secure low-latency access to data.
- Reduced cloud ingress/egress costs.
- Easily adapt to regulatory change.
- Data can be governed by policy.
- Minimized risk of data loss, data leakage and data theft without compromising accessibility.
- Store petabytes of data across multiple locations, with a single global namespace (cost-effectively).
- Remain in control of the data at all times.
- Cloud agnostic data services with secure multicloud access.

Edge Node Deployment

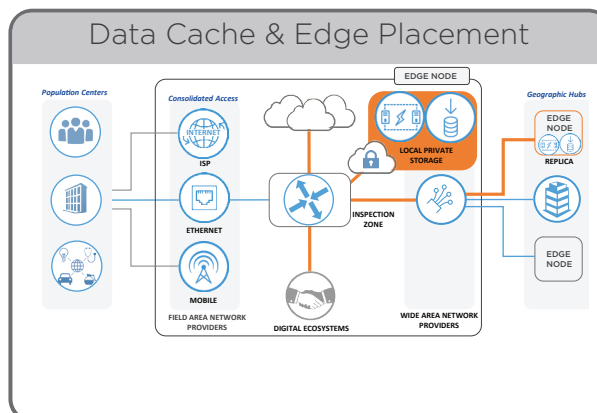


Implementation is reflective of the amount of data processing. As a guide, 1PB of data usually is being serviced by 5 cabinets of IT gear.

DESIGN PATTERNS

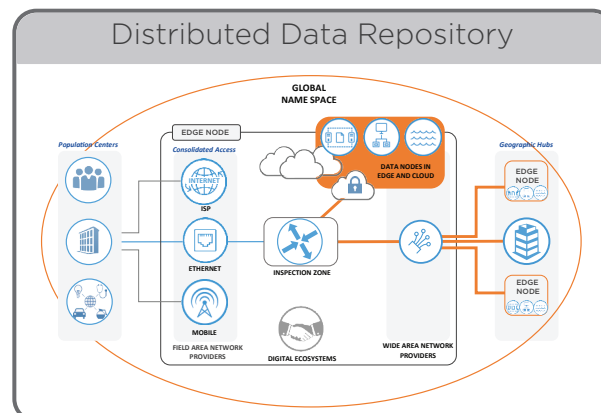
1 STEP 1

First, address data performance and data sovereignty by placing data caches/copies in proximity, or keeping the data local. Also solve for multicloud access by placing data at network intersection points.



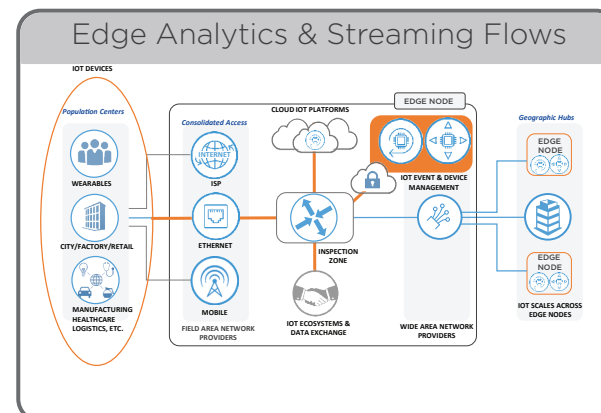
2 STEP 2

Next solve for data availability by placing a global namespace over the edge nodes and extending that into the various clouds as appropriate.



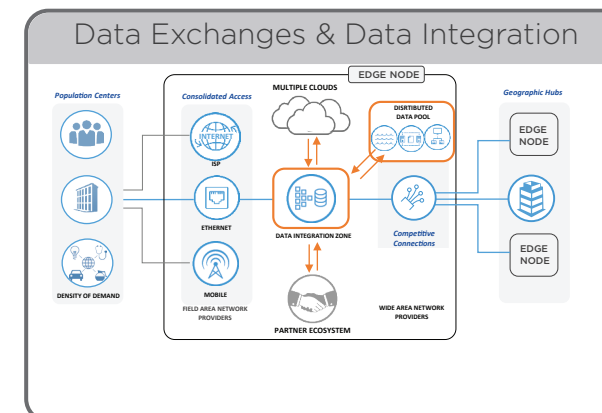
3 STEP 3

Leverage edge nodes for data collection and data aggregation, with local event processing to optimize streaming and time to insight.



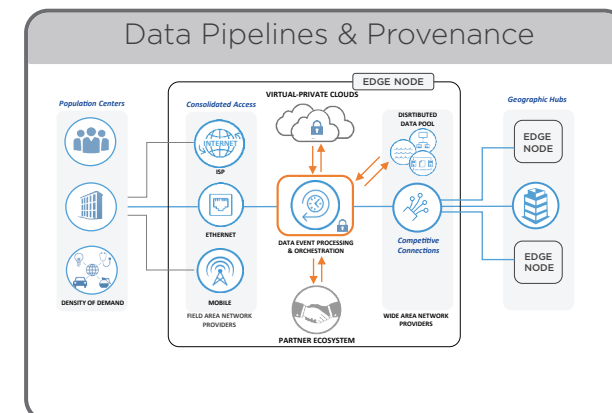
4 STEP 4

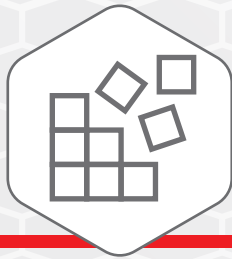
Combine data from disparate sources into meaningful information and deliver it as trusted data (which can be monetized and exchanged).



5 STEP 5

Lastly, you will need data orchestration and data provenance to facilitate and track data flows and consumption from disparate sources across the data fabric.





Problem

As workloads move to different clouds and to the edge, accessing centralized data over WAN introduces unacceptable amounts of latency. At the same time, there are reservations to copying the data into the cloud (ingress/egress costs). This leaves most at an impasse—slowing cloud migration.



Solution

Deploy local data services into the edge node and leverage the low-latency, high-throughput, direct cloud connectivity (Network Step 3*) for access. This way the data can be accessed by applications from multiple clouds, partners, or from users coming in over the field area networks (however, it is not actually stored in the cloud). This private storage at the edge provides a different tier of performance than the data repository (Data Step 1). The type of implementation will depend on access volume, frequency and data size (in memory, solid state or hybrid) with asynchronization capabilities. Typical use cases involve a mix of data types—images (VMs, containers, application binaries and libraries), media content and workload data sets (e.g., batch data for a compute farm, etc.). Most providers have tools for centralized management (for this environment, a service API is also needed). This typically supports more block interfaces (FC, FCoE, iSCSI, NFS, pNFS, CIFS/SMB), and today an object/API.



Constraints

1. Moving large datasets out to the edge can create management and accountability problems. Current infrastructure was not designed to support that and it can fall off the radar.
2. Workloads have been architected and sized based on local I/O expectations. Moving the workload without the data is rarely feasible.
3. The data is mostly unclassified. It might be able to go to cloud, or it might not. To move it may need CIO approval, which definitely slows down cloud migrations.
4. Traditional infrastructure had tightly coupled services, such that storage included backups, snapshots, off-site replicas, etc. Datasets in the cloud require the coverage.
5. Cost of storage is frequently misused in these discussions, which can prevent exploring solutions and alternatives.



Steps

1. Deploy the local private storage into the edge node. Add a second instance if you need failover/recovery. The second instance could also reside in a different edge node.
2. Replicate the nodes to each other.
3. Attach interfaces to the segmented networks they will be servicing (including cloud access).
4. Integrate with boundary control and the inspection zone (Security Blueprint*).
5. Register with the vendor(s) management tools and publish a self-service API.
6. Configure policies and integrate with policy management. Collect events and logging.
7. Configure daily snapshots.
8. To back it up, leverage the object interface to backup directly to the distributed repository (or direct to cloud storage (Step 1)).
9. Configure data to be pushed to it if being used as a cache.



Forces

- Data needs to be moved closer to the edge, where business and customer engagement occurs.
- With changing regulations and increasing reports on data leakage, sensitivity levels to data breaches are high and the rules on certain data types can also change.
- Business is moving lots of functions to SaaS, which at times means having to encrypt and mask the data being stored.
- Storage concerns drive some renegade behavior (shadow IT) which, right or wrong, can expose the business to risk and contribute to data sprawl.

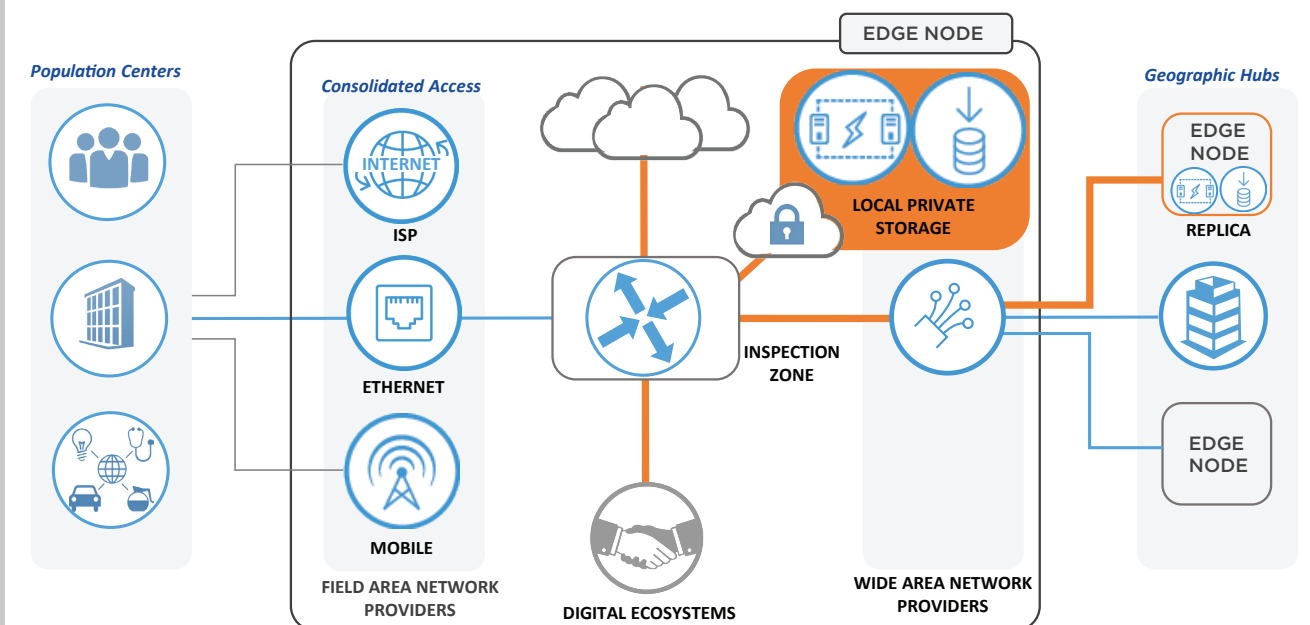


Results

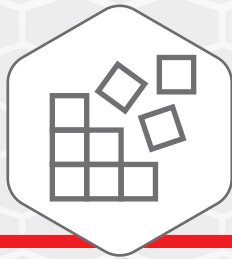
- Localizing data removes the bulk of the latency and is best positioned at the intersection point.
- Running multicloud application workloads doesn't require moving data — access the data in the edge node over secure, low-latency connectivity.
- You have added layers of data protection so it scales without increasing risk.
- Can be used to migrate data between clouds.
- Act as a data exchange server for access to monetized data sets shared with partners across segmented networks.
- Additional streaming data and real-time data cache tools can use this service as a backing store, de-staging lower priority data.



Reference View



* Network and Security Blueprints — IOAKB.com



Problem

The complexities of large-scale datasets (in the petabytes and growing) exacerbates problems with concentration risk, and QOE suffers. Without reasonably performant distributed access at the edge, and tools to segment access logically and geographically, the pendulum swings back to distributed data sprawl.



Solution

Deploy a single namespace data service that is available in all edge node locations, optimizing for high availability and data protection. It can be immutable to protect against human error and data corruption, and provide policy-based controls to address logical and geographical data segmentation. This is the same proven technology cloud providers use to support massive scale, multitenant data services (i.e., private cloud storage, object storage like S3, etc.). It has not been widely used by enterprises since it needs multiple geographical locations and an optimized WAN to be truly effective. IOA solves that. Geographically place data nodes in each edge node (and cloud environment(s)). From there, built-in algorithms interpret policies and store the actual data in a way that protects from device, location or even regional failures without losing data or access. This offers far more protection than a "copy" and uses much less storage. Data services are also optimized for integration, supporting multiple interfaces (web, APIs, file system, etc.). The technology is not new, it's a data abstraction layer across underlying technologies realized with IOA.



Constraints

1. Centralized data architectures solve for localized data requirements at medium scale (by today's standards), and were not designed for geographically distributed access at a scale well beyond current design limitations.
2. Data sovereignty and privacy regulations force the geographical localization of some data types. These same concerns make cloud as an alternative controversial, or not an option at all.
3. Industry compliance requirements also limit what can be done with data.
4. As businesses and applications struggle to shift to the edge, the management capabilities needed to bring the data are not in place.
5. New data is continuously being generated at the edge, either creating more sprawl, or experiencing a bandwidth backhaul problem with ongoing capacity and its inherent risks.



Steps

1. Design the placement of data nodes. Implementation can vary; however, a best practice is to have a minimum of four locations (cloud environment virtual machines count) and for optimal protection 16 nodes (four locations each with four nodes).
2. Place the first data node and start the service. As more data nodes are added, your private data cloud/namespace will expand in size and the service updates itself.
3. Integrate the service with boundary control, inspection zone(s) (Security Blueprint*), policy enforcement and API management (Application Blueprint*).
4. Apply event processing and monitoring.
5. Establish logical capacity buckets and access groups with protection and placement policies.
6. Continue to add more data nodes to scale namespace capacity. Data nodes deployed in the cloud can be backed by cloud-provided storage (e.g., S3). Policy determines cloud use.



Forces

- Data is growing — from GBs, to TBs, to PBs and now approaching ZBs. Proportionally more data is being generated each year than what was already stored.
- Data is no longer centralized. As traffic and processing have shifted to the edge, data growth and consumption are being drawn with it. The physics of latency along with ever-increasing bandwidth costs mean that backhauling all this data is not sustainable.
- Data is becoming more valuable — to the business, to customers, to competitors and threats. Businesses that were product driven are now going to be data driven.
- This drives the need for data services that provide data availability at the edge and accommodate logical and geographical data segmentation.

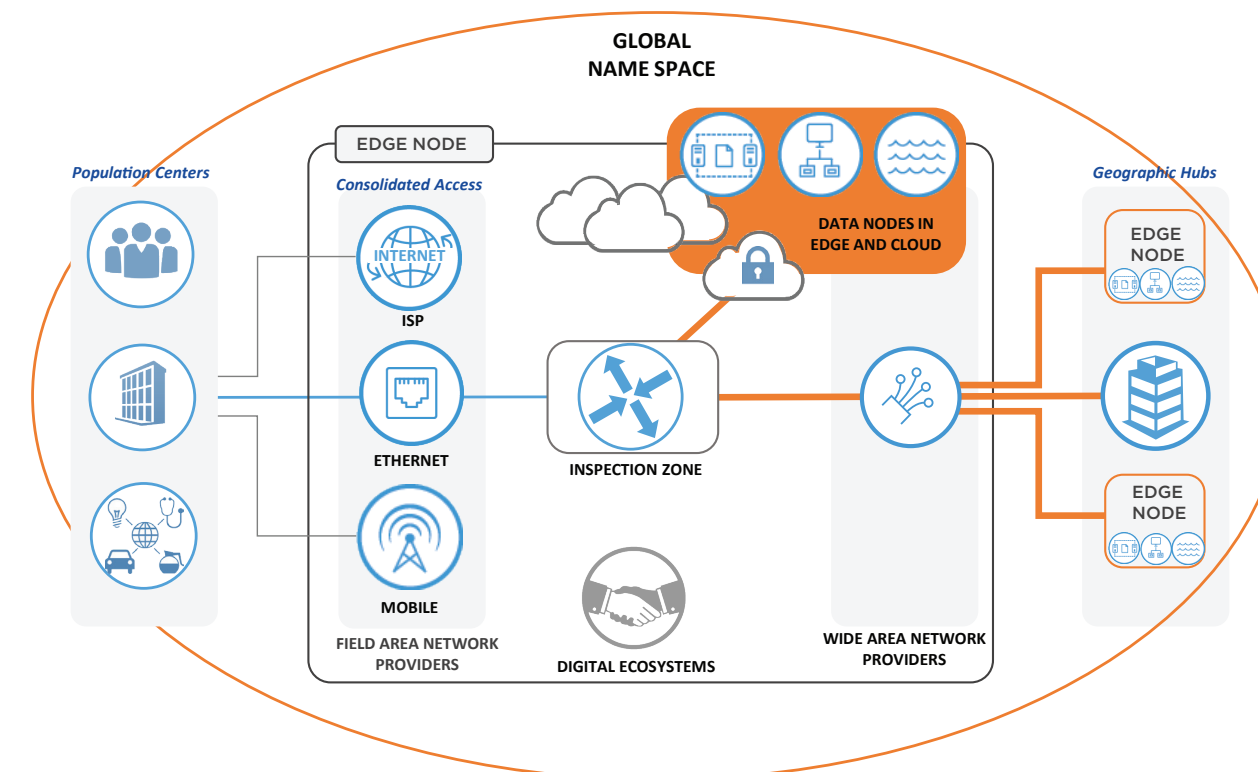


Results

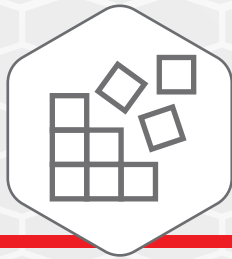
- A distributed service with centralized management solves two previous constraints.
- Designed to handle zettabyte scale data sizes.
- Accessibility reduces the volume of data moved and copied—and therefore stored (disk space) significantly, which reduces WAN bandwidth and storage costs.
- All data can be automatically encrypted at the data layer (the key is also dispersed and not stored in any one place).
- With security and policy management integration, ensure compliance consistency in all regions.
- Any non-latency-sensitive data requirements use the repository as primary storage and archive ("eventually consistent" regionally).
- Examples of use include shared drive, package distribution for apps/containers, logging repository, staging area for analytics, etc.



Reference View



* Security and Application Blueprints — IOAKB.com



Problem

Collecting data from the field (e.g., IoT) is more complex than traditional analytics. It's intermittent, highly unstructured and typically real-time. Transferring all that data into cloud incurs latency, and bandwidth won't scale with the growth in volumes and data. Real-time analysis becomes less achievable.



Solution

Place IoT event processing and device management (firmware and updates) in one or more of the edge nodes. Since the node is located at the intersection point of consolidated access (cellular, broadband, internet, etc.)—including frequency networks (you install as more bands become available)—the edge node is the closest point to the field and the clouds. Placing device management and IoT analytical capabilities at the edge solves latency, bandwidth and device complexity constraints. It also provides multidestination control and choice in the types of network/providers you want to use, as well as which cloud analytic platforms are available. Participate in IoT ecosystems by connecting with partners and exchanging data. As data comes in it is validated, authenticated, inspected, pre-processed, stored in the global namespace, and then delivered to the next downstream processing step.



Constraints

1. Data from the field isn't the same as prepared data — it can be messy, intermittent, unstructured and somewhat dynamic (if the device is mobile).
2. The volumes of devices, variety of sources, frequency of samples and data are all growing, which will not scale using traditional approaches.
3. In many use cases latency matters, and delays between the event and the reaction need to be kept to near-real-time. Delays will become unacceptable as throughput is impacted by that growth.
4. Traditional approaches to centralizing all the data to run analytics (in the cloud) are not sustainable for real-time use cases. But in this architecture, where else can it be done?
5. Alternatives that put more machine learning intelligence in the device increase device complexity, draw more power and lead to the results in data being discarded. When we decide we need that data, we are back to where we started.



Steps

1. Establish segmentation flows from field area networks. Messages from IoT gateways (in the field) get published on the message bus; otherwise a local IoT gateway processes it first.
2. Boundary control validates and authenticates the source and message. (Security Step 1).
3. Valid messages go through an inspection zone and policy enforcement (Security Steps 2 and 3).
4. Messages are persisted to the data repository (Data Step 1), then delivered IoT event processing. Downstream messages are published to go onto cloud analytic platform(s) of choice, or your own cloud-agnostic repository.
5. IoT gateway "device requests" go through the same flow but are subscribed to by the device management function. Appropriate payload(s) (firmware, etc.) are loaded from the data repository and published back to the IoT gateway (or device directly).



Forces

- The growth of traffic and data at the edge is driving the need to reduce distance and bring analytics closer. Distributing computational processing is a known practice (data gravity).
- While bandwidth is cost prohibitive, latency causes the biggest problem, as there is a point where adding more bandwidth only marginally helps (physics).
- IoT and related field use cases are driving more simplicity into the field from the edge. It's easier to change something in 10 places than 10,000,000.
- Likewise, not all actions required on all data are equal, and investigation into staging processing allows a more balanced approach. Level 2 data does not need the heavy processing that Level 5 does.

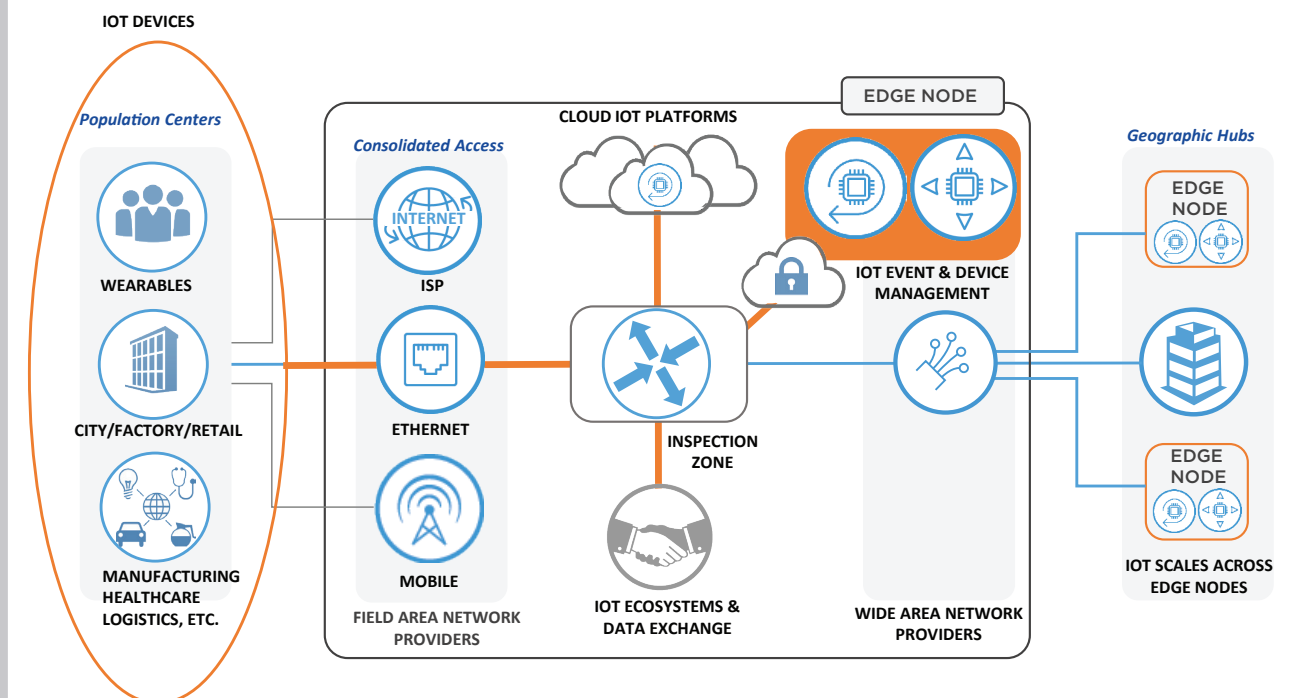


Results

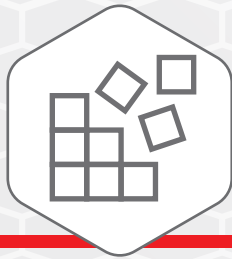
- IoT edge capabilities that can scale to billions of devices, at each metro location globally.
- Collected data can be validated, authenticated and inspected before being pre-processed. (Security Blueprint).
- All data can be stored in the global namespace (Step 1) — no need to discard.
- Most efficient use of bandwidth with lowest latency for real-time event reactions and the most efficient way to scale.
- As much processing as needed can be localized at the edge, with choice of cloud IoT platforms.
- Monetize data and sell access on a data exchange, gaining the insights and generating revenue.



Reference View



* Security Blueprint — IOAKB.com



Problem

Access to multiple types of data from numerous sources and locations in either an event-based or time-scheduled manner places a large burden on enterprise data infrastructure and services.



Solution

Deploy a data integration platform in the edge node. A data integration platform essentially takes data sources from a number of supported source interfaces (file, database, object store, etc.), transforms it into a universal format, and then uses data services to provide varied consumer interface choices in order to consume the data. This already has widespread value to organizations that frequently need to integrate data between disparate applications (in one or more clouds). Used a different way, this platform also enables a data exchange. Data exchanges are groups of companies that are securely interconnected in the edge node for the purpose of accessing/sharing data (which is typically monetized). New data sources are valuable to data-oriented partners. As in analytical processing, more data sources directly translate to more experience (it could be IoT data, scientific data, medical trial data, etc.). Even if "translation" is not required and data is passed straight through, the other governance functions provide significant value and needed oversight in a dynamic, automated environment.



Constraints

1. There are hundreds of permutations of data transfer occurring. Some of these require governance items that may not be applied.
2. In a "trust nothing" environment, each action that changes the data should have some level of governance review, which is impractical for individual service implementers to do — anyone can call their service from an API (dynamic with no humans involved).
3. Random acts of data transformation (in some form or level) exist throughout the environment. Services like ETL (extract, transform and load) consume resources and lack macro coordination.
4. Data guardians experience difficulty governing the dynamic transformation, yet are still accountable for the data.



Steps

1. Deploy access (adapters/connectors), transformation and delivery services (adapters/connectors).
2. Wire each step to go through boundary control and inspection zone(s) (Security Blueprint*).
3. Apply event processing and policy enforcement.
4. Configure data profiling, data quality and operational processing.
5. Provide internal and external (productized) APIs for data integration (as a service).
6. Configure data repository (Step 1) for data staging and local private storage (Step 2) for caching/performance.
7. Apply metadata and master data, management and re-encryption for destination key management integration (Security Blueprint*).
8. Integrate with a data pipeline service and update provenance information.



Forces

- Data will soon replace traditional products in most firms as the most revenue-generating asset (digital economy).
- Universal methods of accessing, passing and transferring data between disparate systems, companies and networks will be a critical capability. As the need for exchange increases, it needs to be balanced with the mitigation of potential risks.
- The dependency on data sources will drive legal and service expectations (with monetary impact being almost guaranteed) in the event of unavailability or data loss.
- As data leaves a domain of control, where it will go, how it will be protected and what it will be used for may need to be policed.

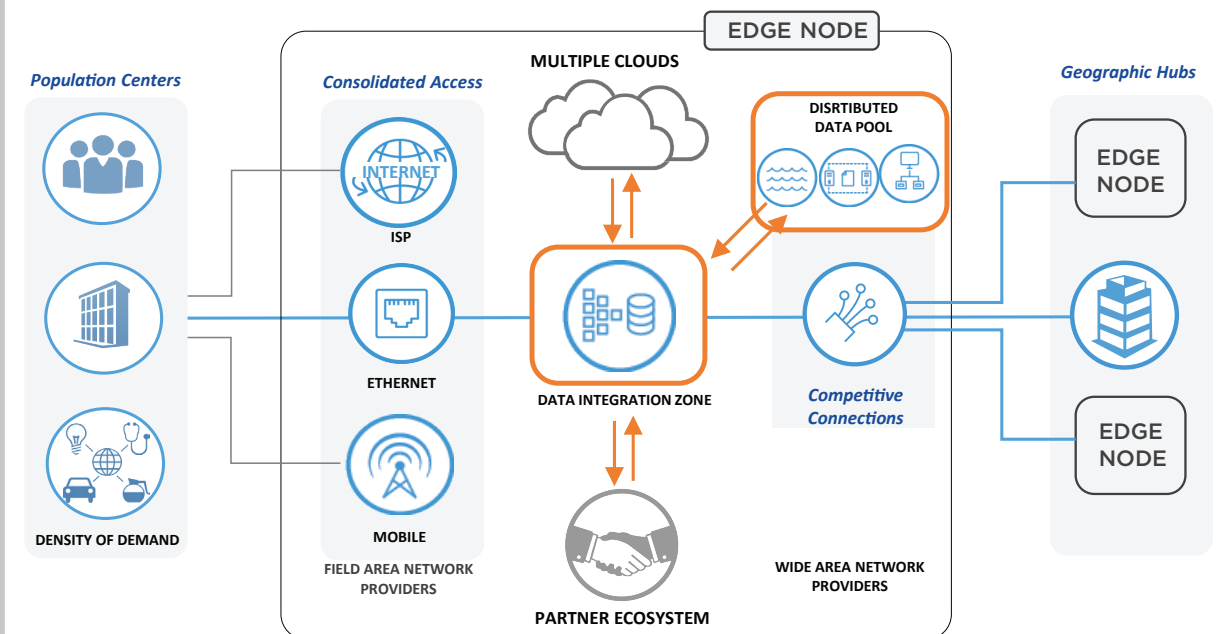


Results

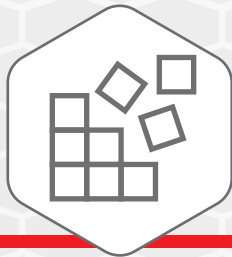
- Uniform way to exchange data between applications, cloud services and business ecosystem partners (with or without compression).
- Delivering dynamically integrated real-time data and managed batch transfer/migrations.
- Full event processing of all data exchange points that go through a data exchange (internally and externally), forming a data events view and audit trail, with dependency analysis.
- Data encryption, masking, PII (personally identifiable information) alerting and leakage prevention can be applied. The exchange is another policy enforcement point.
- Detect data corruption and tampering with machine learning (Application Blueprint*).
- Data exchanges allow you to produce entirely new business models built on integrated data.



Reference View



* Security and Application Blueprints — IOAKB.com



Problem

As more applications become more API-centric and are assembled (and re-assembled), business context and output value take the form of data (small and large). Coordinating and following what is essentially a series of data value chains requires an over arching view that is not in a document, but handled such that it can be monitored, check pointed and even dynamically updated.



Solution

Publish APIs using API management (Application Blueprint*) that process information about data activities and which are then called by (or bundled into) application and service APIs to facilitate automated metadata management. Since, at the time of the event, the application has the information, auto update it as it happens. In addition, data movement and coordination requires over-arching orchestration that can be subscribed to. This could simply be a valet interface into the integration service. Next, leverage event processing and monitoring throughout the platform, including pre-processed service views summarizing activity in their respective domains of control (boundary, inspection, policy management, data services, data integration and API management). For security, data and application design patterns relationships can be auto determined and combined with the metadata. Patterns of data activity can be created as views with a dashboard or by observing data interactions. While it is encouraged that all data interactions use the data integration service (data pattern), even with extreme low latency, that level of introspection may be complete overkill for the task at hand. In those cases, APIs updated to metadata should suffice. At this point, this mega-pattern allows you to deliver an automated self-updating view of all data movement inside the environment and across clouds and ecosystems.



Constraints

1. As a whole, data flows and their associated metadata information are rarely documented or maintained.
2. While someone knows how the choreography of the application flow works, people change roles or leave and the knowledge goes with them.
3. Data movement, transfers, feeds, ETL, versioning, etc., all impact the health and performance of business operations but are not as visible or operationalized.
4. Therefore, consistency in data protection, security, quality, etc., may be lost as information traverses the environment.
5. Introduction of an ecosystem of partners and fast-paced changes to standup new business models quickly increases risk.



Steps

1. Apply data orchestration services to add scheduling and coordination of the other data services (listed below).
2. Leverage the data repository (self distributes globally) and local private storage (snapshots and access changes) for large data sets.
3. Leverage the data integration service for data services and data translation as needed.
4. Establish the provenance function to keep track of and publish APIs for automatic updates of metadata management and to query the service.
5. Leverage complex event processing (Application Blueprint*) to learn data relationships and trails, as well as construct views on data activities.
6. Update policy enforcement to flag anomalies of rogue data access and movement.



Forces

- Controls must be balanced with performance requirements for speed of access and data transfer.
- Metadata information will become almost impossible to maintain (in rate of change and transparency and understanding).
- As the size and overall expense of data continues to rapidly grow, details on who is generating and using data will need to be well understood. Furthermore, its use needs to be correlated to business processes and value/benefits as well as inform company risk profiles.
- The proliferation of analytics, the demand for more data, and the ease at which data-driving functions are being cloned are forcing operational considerations to be solved first.

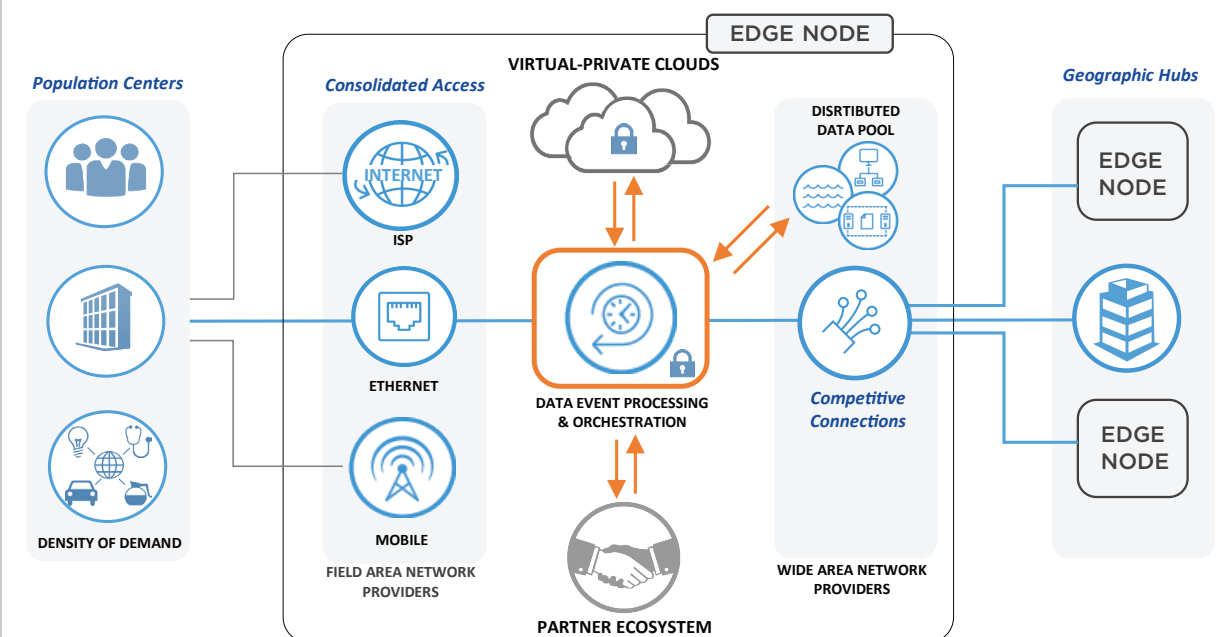


Results

- As an integrated family of optimized functions, the arduous challenge of master data management, risk and data security is made easier — without compromising availability, accessibility or performance.
- These services can be integrated with cloud services such that cloud use seamlessly satisfies operational requirements, updates dashboards and provides insights.
- Should a data breach occur — or more likely a colossal mistake — automatic checks and balances apply protection, with recovery as a fallback.
- Data expiration and HSM have not been covered, but are important. Much of what should be deleted based on policy or migrated to long-term archival is already there.



Reference View



* Application Blueprint — IOAKB.com