



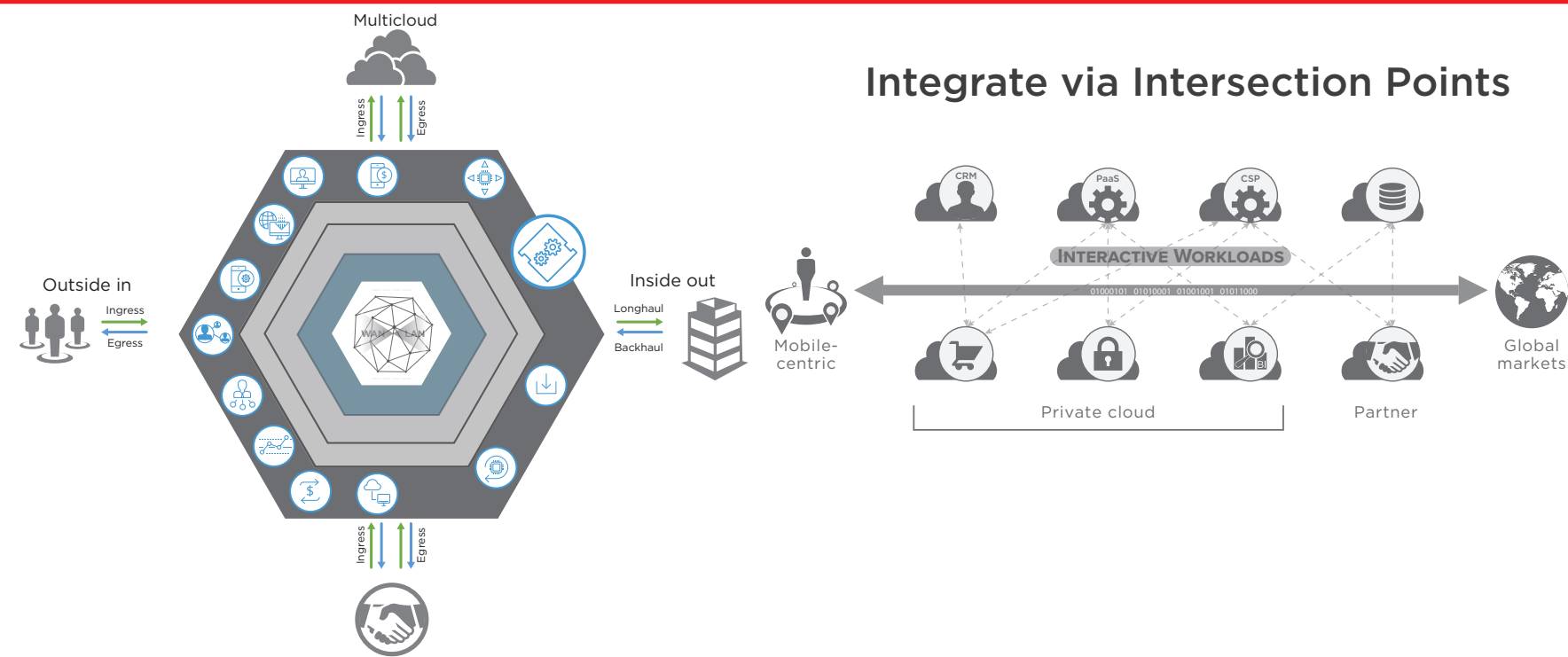
Design Principles

- Business processes are a collection of internal and external digital services.
- APIs are the business interface.
- Applications are interconnected APIs.
- Flows use common messaging over secure and segmented interconnects with policy enforcement (and event processing).
- Systems provision, configure, connect, control and tear-down services (not people).
- Algorithms process data according to models and identifiable patterns.
- Leverage the ecosystem for solutions first.

Edge Node Components



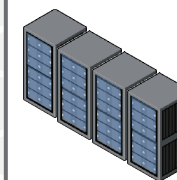
To architect for the digital edge, you need to localize application services in the digital edge node as a multicloud, multiparty business integration point. The shift to digital is trading complicated monoliths for complex interacting systems of digital services and transforming application development to become API-centric in building inter-networked components. Each edge node is an application communication gateway, as well as a place to colocate application functions with latency or volume-driven workloads (i.e., improving user experience, etc.).



Capabilities

- Measure and manage API usage and performance throughout the API life cycle.
- Manage secure application messaging flows end-to-end over secure, segmented, low-latency interconnections.
- Scale distributed services globally and across multiple clouds without increasing management overhead.
- Identify meaningful events from network intrusion to fraud detection, and anything about traffic that traverses the edge node.
- Use algorithms to start learning how and where business capabilities can be improved; in the field or across partner exchanges.

Edge Node Deployment

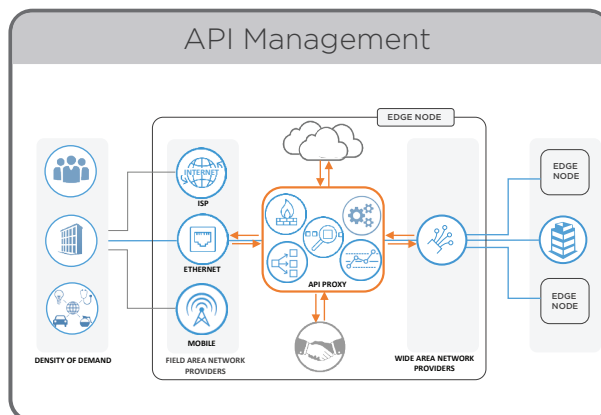


For those services placed in the edge node, your ratio of apps/containers needs to be applied to determine footprint planning.

DESIGN PATTERNS

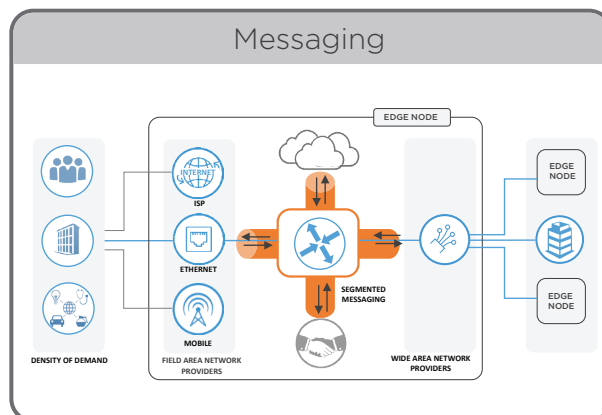
1 STEP 1

The digital storefront is shifting from websites to APIs. You need a managed clearing house for APIs that you are producing, publishing and also consuming—placed in the intersection point.



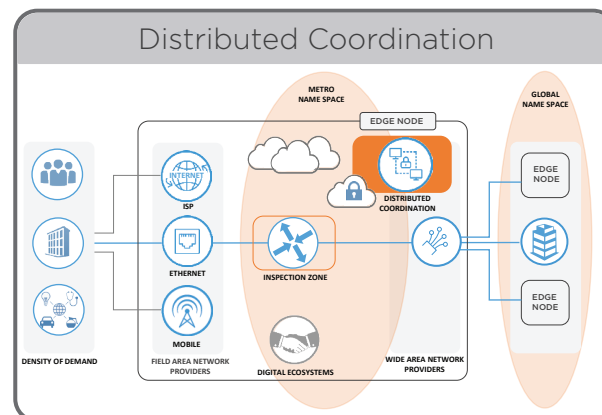
2 STEP 2

Plumb your messaging infrastructure for application flows that will traverse the edge node, aligned with traffic segmentation.



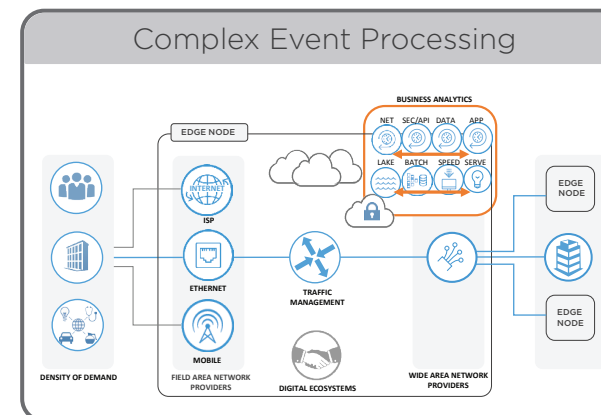
3 STEP 3

As components become increasingly distributed across clouds and edge nodes, you need to scale your distributed coordination and configuration capabilities accordingly.



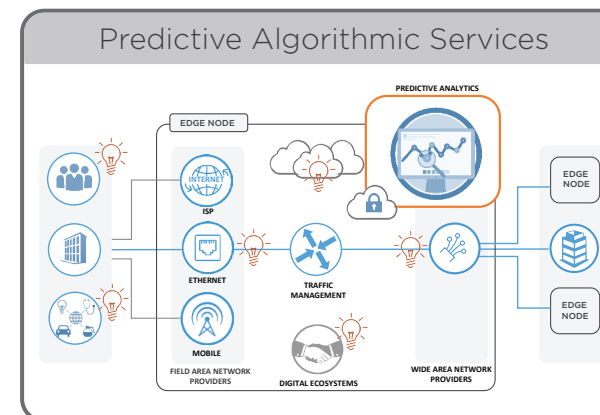
4 STEP 4

With event processing in place at the network edge, intersection points, security checkpoints, data services, APIs and messaging — you need to the capability to manage the bigger picture.



5 STEP 5

After you can model the bigger picture, your digital platform needs to start learning and recommending, along with taking preemptive actions.





Problem

As products shift to APIs, these business interfaces could be spread across multiple clouds and PaaS environments. Coordinating the organization's digital brand and end-to-end service levels is challenging.



Solution

Leveraging the architecture from the Network and Security blueprints, establish your external API product platform and deploy API management functions to the inspection zone in edge node(s) to optimize consumer experience. This is, in short, a proxy service (local or SaaS based) for business API access at the digital edge. The inspection zone is optimally placed at the intersection point for all segmented traffic flows, and is therefore the closest point to all consumers, partners and clouds. API requests may come from some or all of those segmented flows and even from inter-metro WAN links as you regionally balance requests. This allows you to configure which external product APIs will be available/consumable by whom, on each segmented network, with policy enforcement and managed SLAs across end-to-end customer experience. While some additional components are still needed, you can leverage boundary and inspection zone services already in place (or look to converged options). Next tailor availability of API products across the other edge nodes and regions.



Constraints

1. A website was, in many ways, an e-commerce façade to a disparate set of backend systems. However, it did coordinate a single common external interface to the world. API-centric products require an entirely different architecture and bypass the website.
2. Multiple cloud providers are providing tools to help develop APIs for business products. This is good for kick-starting development of internal-facing APIs, but a corporate strategy needs to be formed first. Where will external interfaces be located?
3. Consumers accessing APIs may have contractual service requirements. In addition, partners may need reseller tools. Will developers be responsible for all the cross-API operational aspects? And will they be globally coordinated into a single story for the customers?



Steps

1. Augment the Boundary and Inspection Zone with API capabilities—either with more SaaS services, or dedicated appliances.
2. One scenario is to place an external API reverse proxy in boundary control (manage authentication, authorization, users, regions, encryption, filtering, etc.) for all APIs, with event processing and an application firewall.
3. Next, inspect authorized API calls, enforce policies and apply event processing (not just logging every call, updating statistics, etc.).
4. "Inspected" API calls go to an internal API gateway (to manage calling downstream internal APIs or publish messages). This layer manages internal versioning, rate limiting, load balancing, etc. Event processing of API calls is also applied.
5. Build all the business (or fulfillment) services behind this API proxy/gateway service with internal APIs.



Forces

- In digital, products and services are delivered within an ecosystem. Value comes from being able to cross integrate functionalities.
- This means your products have rich APIs with a great partner/consumer experience, which in turn make it easy to do business with you.
- Since business is now about APIs, business operations management, product management and customer service are expecting APIs to be managed like a business.
- DevOps (hype aside) is about developing new functional products, with built-in operational management and service instrumentation and practices.
- Business performance will come down to overall and end-to-end API execution and customer service.

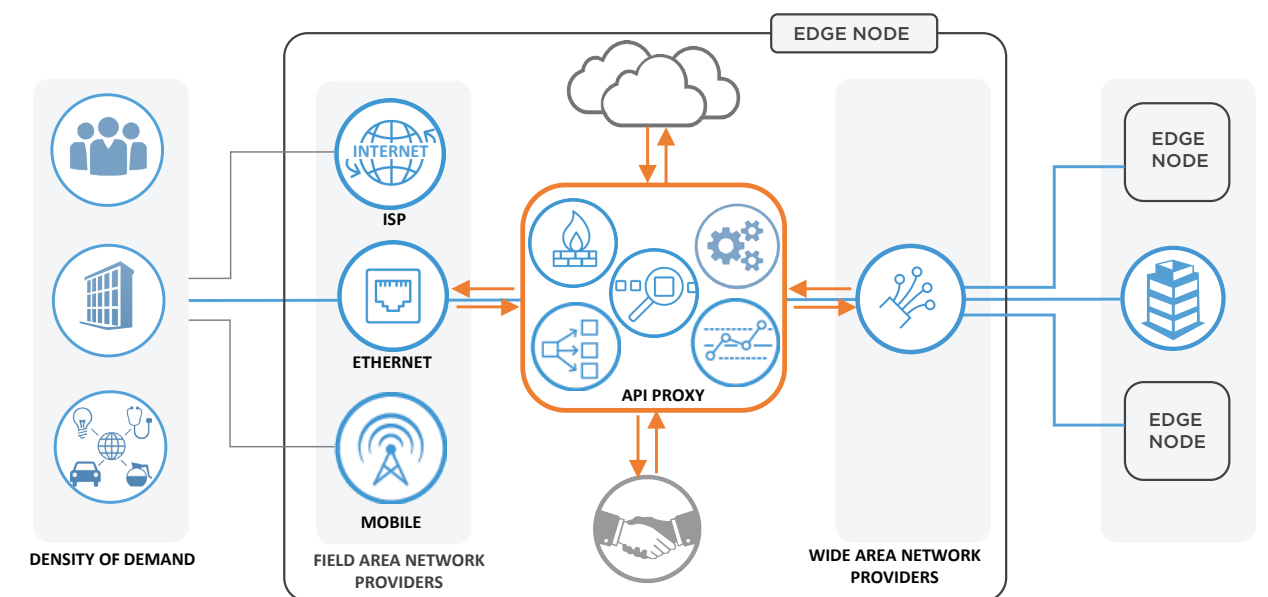


Results

- Shortest path to business product APIs with lowest localized latency and most efficient bandwidth — maximizing throughput.
- Leverage the intersection point(s) as an optimal place to observe business communications at multiple levels in the stack. Provide E2E SLAs.
- Develop to versioned internal APIs that are registered with API proxy. External APIs in boundary control are more stable (lower tolerance to change).
- Migration from monolithic apps to micro services happens behind the scenes.
- Cloud services can be integrated without performance impact.
- Stamp out designs across other edge nodes for global business platform.



Reference View





Problem

Even with a common API strategy, the environment is a hybrid mix of applications and multicloud services that alter between synchronous and asynchronous behaviors. What does the platform do about integration across increasingly disparate and distributed services?



Solution

Optimizing for application (service) integration first, and leveraging the edge node's secure communication gateways, plumb the edge nodes with messaging services that can extend into, or connect with, multiple cloud and SaaS environments, business partner networks and services, and inbound messages from field area networks (like MQTT for IoT). API Management (step 1) has solved for modern API/REST architectures, but this step also solves the reality of integrating the rest of the environment, delivering a scalable runtime platform optimized for localized services across low-latency cross connections, where you can afford to add additional services without taking a performance hit. Likewise, many services offer powerful connectors that provide what you need out-of-the-box (e.g., SAP). Either way, regardless of developer, cloud, or partner philosophy or maturity—all can be easily integrated.



Constraints

1. Not all applications are new and may not be able to be changed to meet new application architectures.
2. In a hybrid multicloud environment, some tribute has been paid to portability (but not much to interoperability).
3. Digital partners and ecosystems will gravitate to a center of density to exchange services and peer business traffic. However, their application platforms will come from completely different architectures, or be further ahead or behind other members of the ecosystem.
4. Technology is moving quickly and that momentum may bring convergence. However, digital business favors who is first. The priority is integration over standardization. This was not a priority in the past.



Steps

1. Determine the messaging support needed for each network segmentation flow type. Focus here on variety and volume.
2. Interconnect messaging with the security and inspection zone flows and business API proxy.
3. Publish an API for the messaging service itself.
4. Integrate data services (as per Data Blueprint*).
5. Create REST APIs using the API management tools (step 1), and/or apply connectors so components are registered as required (take the path of least resistance and speed).
6. Implement event processing (and correlation if applicable) and logging.
7. Coordinate/integrate with policy enforcement.
8. Integrate with data pipeline management.
9. Configure messaging translation to rationalize (reduce) protocols in order to optimize inspection and gateway capabilities.



Forces

- Businesses are looking to integrate services in order to achieve new business models, but the digital economy cannot wait for participants to standardize applications and protocols. Time to market drives business value.
- The number of data sources is also increasing in the multicloud environment (CRP, ERP, B2B, Hadoop, streaming, databases, and now IoT). They also all need to be integrated (as per Data Blueprint*).
- Micro services are promoting intelligent endpoints and simple messaging integration—other services still require messaging to handle the integration seamlessly.
- In any event, both agree integration is the most important objective.

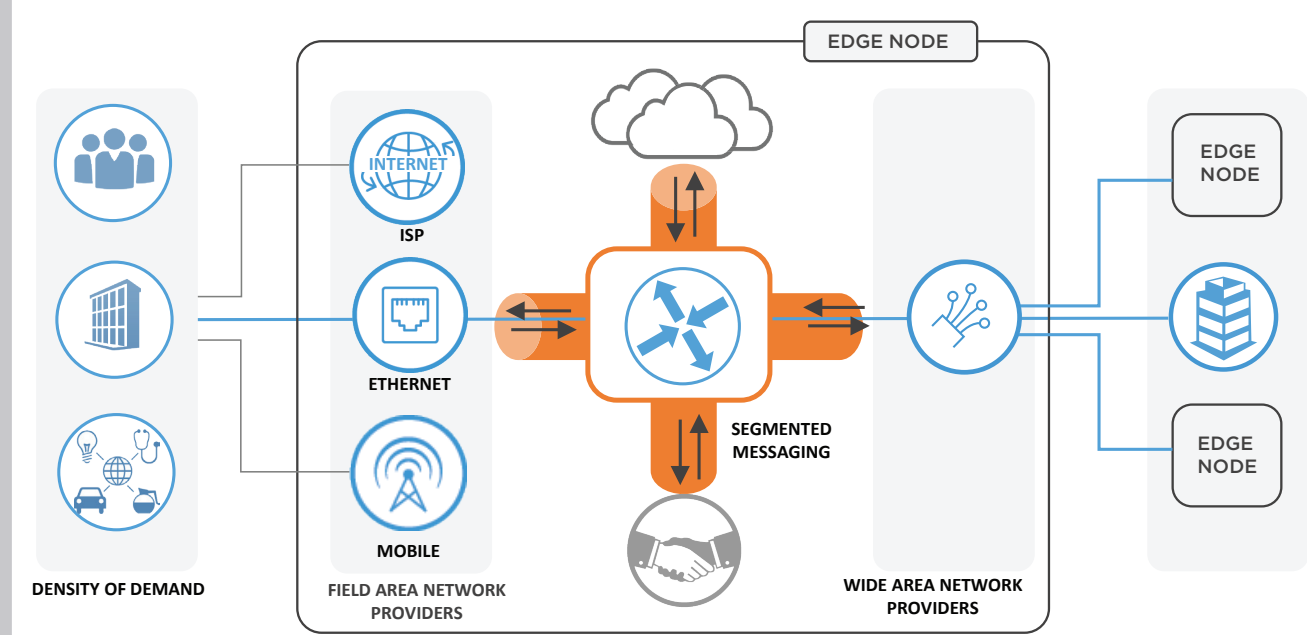


Results

- The edge nodes and interconnected networks now have broad communication gateway support at all levels of the OSI stack.
- Edge nodes can easily be sized for extreme volumes (like IoT) with the throughput needed to scale millions of localized messages and events without compromising security.
- Primary data services (Data Blueprint*) are discoverable and available. More can be added (external sources or feeds) and policy controlled.
- Cloud services are, or can be, integrated as necessary to support hybrid deployments.
- A platform designed to integrate first, with a wide range of options to make it easy for business partners.



Reference View



* Data Blueprint — IOAKB.com



Problem

How will run-time configuration state be coordinated? Not just across internal business services, but also across other disparate services that may or may not be otherwise connected to a common messaging platform.



Solution

Add distributed configuration and coordination capabilities (DCC) to the edge nodes and make them available through your platform, addressing this particular integration challenge. The implementation can be extended into partner and cloud environments and made available across those segmented networks. While other approaches can coexist, this service provides the universal way to solve widespread application state coordination where it is needed. This implementation is typically a secure, distributed, in-memory namespace that application affinity groups use to share values and state information. Its scope includes distributed run-time application configuration and coordination, which does not have a large amount of data involved, and as such fits alongside messaging and API gateways. Additional capabilities could be added to handle stateful connection tracking.



Constraints

1. Each application environment has already made and implemented its answer to distributed coordination—which could even be to not do it at all. In any case, the goal wasn't end-to-end coordination as a common solution—creating a gap.
2. The shift to API-centric direction in application development has somewhat relaxed decisions around what technology and approach is used behind the API (including programming languages which determine tools, etc.). This creates additional disparity in implementation.
3. Even if there were common and ready services for developers to use, there is no onboarding process, documentation or test environment to simulate what the integration platform looks like. Who owns cross-application integration as a service?



Steps

1. Optimize DCC for an implementation that has the broadest support and easiest path for consumer adoption.
2. Interconnect DCC with security services and access through the inspection zone.
3. Publish a service API for the DCC (add, change, del) and connectors.
4. Use the API to establish secure namespaces for components and take over their ongoing configuration management (configuration for: network, security, data mappings, message queues, etc.) starting with edge node services.
5. Implement event processing (and correlation if applicable) and logging (service usage, etc.).
6. Coordinate/integrate with policy enforcement for guardrails.



Forces

- The shift to API-centric direction in application development has relaxed the emphasis on what you use to build behind the APIs (including languages and tools used, etc.). This disparity in implementation can add complexity when distributed coordination across those APIs is needed.
- Real-time integrated business flows are increasingly becoming the norm (e.g., retail coupons, digital payments, with online credit and fraud checks – all in less than a second).
- Digital is also driving extreme geographically distributed transactions that may still need transactional integrity (e.g., logistics businesses needing to roll back a transaction across multiple partner functions, etc.).
- Application architects are beginning to have to think about widespread integration at scale.

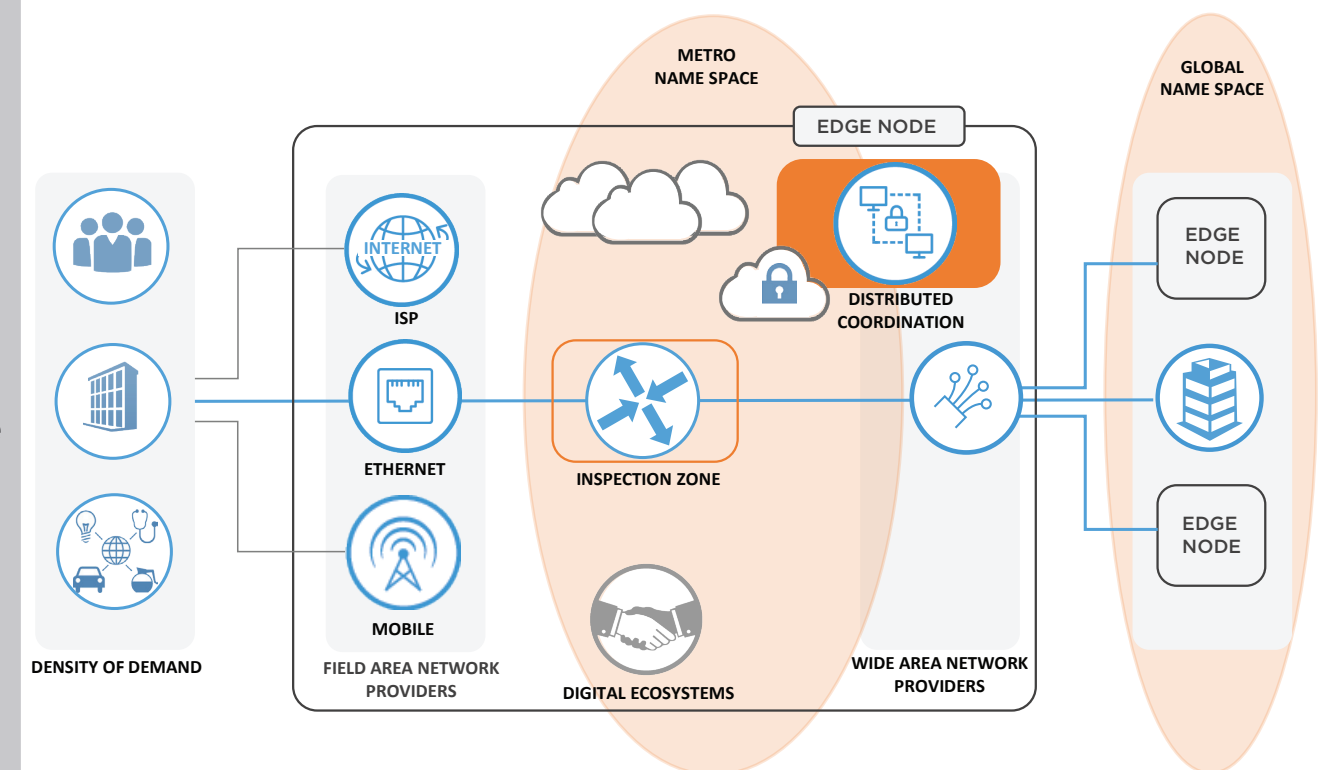


Results

- You have a secure, self-service, distributed configuration and coordination service with mixed local and global groups as needed.
- Implementation across the mesh of edge nodes either as local appliances or SaaS augmentation.
- Components can be parked when provisioned and then assigned their functional configuration. You may not want a faulty component to immediately go back into production. Configure a replacement.
- Administrative tools interact with the namespace for higher level orchestration.
- With the infrastructure-as-code, the namespace can be backed up to the distributed repository (Data Blueprint).



Reference View



* Data Blueprint — IOAKB.com



Problem

We have a collection of event processing and logging functions occurring at networking, security, data and application layers. While that allows for siloed service management, how do you provide a business view of what is happening?



Solution

Provide the enterprise architects (or business technology officers) with a complex event processing platform that is pre-loaded with all the insights from the event processing network, security, data and application services. Models can be defined that represent business operational views backed by transparency into the underlying aspects. These models can be real-time dashboard views of business transactions, with secondary time-based trend reports. Business service levels can be reported on and controlled (to avoid penalties). Optimization teams can identify and investigate the weakest links. Determine where the highest time or resource costs are, benchmarking alternatives in a business context to compare the net business improvement. New business models should be translated into similar analytical models, and published and reported on in real time and batch. Each node would contribute its localized view and share a global view.



Constraints

1. Analytics has emerged as the only practical way to find proverbial needles in the haystack. However, this presents the biggest limitation. Our microscopic viewpoint defines a siloed perspective, and therefore isolation reality.
2. When we try to manage a business scenario, which crosses multiple technology disciplines, we tend to find that each individual service appears to be working fine, but the business experiences a failure.
3. Furthermore, what we are measuring and monitoring is usually not even what matters from a business perspective (e.g., Alert: Jane D. is experiencing a 20% reduction in CoolApp performance; this will cost the firm thousands in lost business and put us on a regulatory radar).
4. When we take these constraints to a multicloud, multi-organization, distributed and dynamically changing environment — running new integrated business models — how well is this understood?



Steps

(leveraging prior blueprint solutions*)

1. The analytical platform components should be a collection of cloud- and SaaS-based services. However, regulatory compliance or data sensitivity rules will require local deployment in the edge node.
2. Apply a data pipeline to aggregate events in the distributed storage repository (data lake) for batch/map reduce processing (batch layer).
3. Apply streaming data services for harvesting real-time processed events (speed layer).
4. Create merged views for analysis (serving layer).
5. Integrate analytics with security and guardrails for the most comprehensive security clearance.
6. Generate service views and dashboard(s) first.
7. Define business views and operational dashboard next.
8. Maintain and fine-tune all the models.



Forces

- The pace of technology and digital business change is increasing, which means that maturity and understanding are not.
- The pace of technology and digital business change is increasing, which is diminishing the overall end-to-end understanding of the environment.
- Automation applies to business processes and more guardrails (policies) are needed to prevent a runaway catastrophe.
- Advanced persistent threats probe on different things throughout the network over longer periods of time — a broader view is needed to detect that several disparate anomalies are actually a coordinated event.
- Technology teams and business units are having to work much closer and understand more about each other's realm.

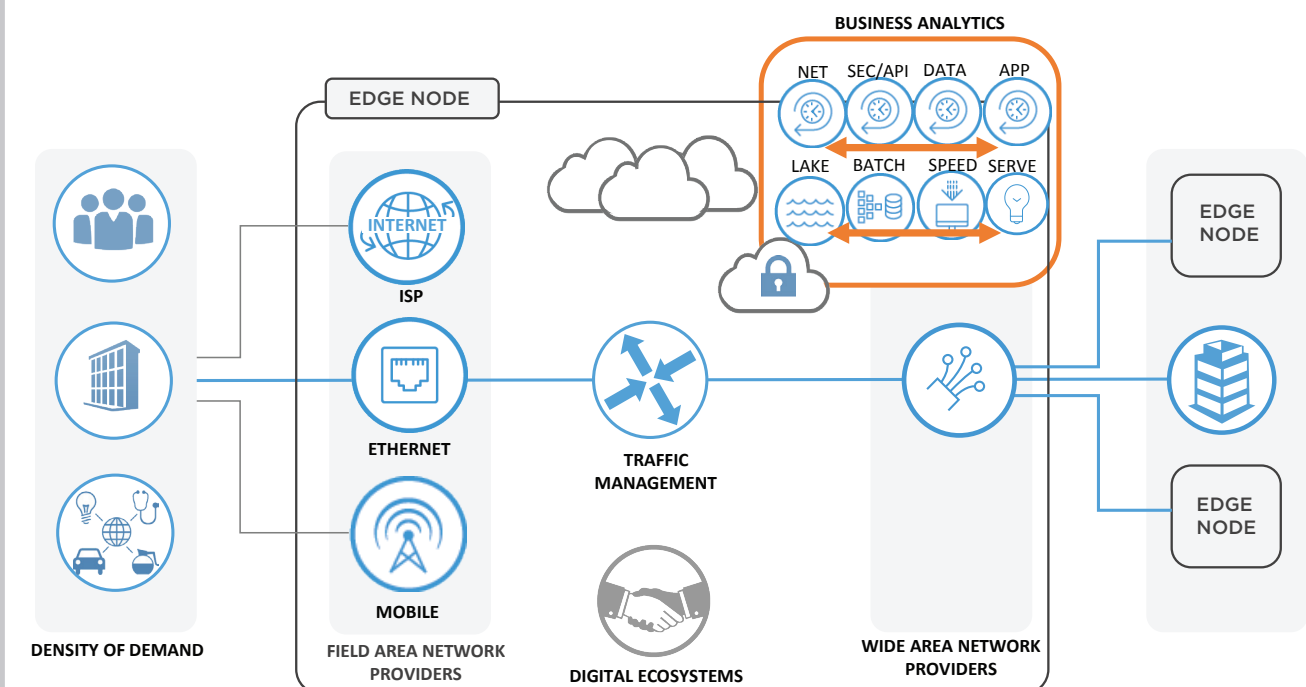


Results

- Everyone should be an expert on what is going on, even if the technology being used is only hours old—introduced to the company yesterday.
- This platform, delivered this way, fully realizes the vision of DevOps (harmony of vertical and horizontal objectives (business and IT)).
- Whether you developed any of the code, or crowdsourced it all, you still have an integrated team of business technology officers whose first priority is rapid integration of capabilities in order to capture new opportunities.
- Every decision made is completely based on current and accurate decision support analytics, and every innovation can be regression tested.



Reference View



* See IOAKB.com



Problem

Even with the best data on what is happening and a deep understanding of how the environment works with controls in place, it is still a reactive model. How can we plan or forecast accurately when everything is constantly changing and we continue to pivot?



Solution

Take the Digital Platform analytic capabilities, built from the steps so far, and now introduce machine learning to apply operational intelligence to our operating models (supported by complex event patterns). Before looking deeply into the data science aspects, the solution here is to broadly apply inductive learning to the analytic models the platform already had. The outcomes will be a mix of predictions around volumes and capacity with relationship discovery (upstream and downstream dependencies). Other predictions are related to business behaviors and the likelihood of outcomes (used today to improve targeting). When put together, you can start modelling what you should do next with actions and implications already determined. Over time, experience builds into the models, which in turn improves the predictions. In short, it's strategic planning at scale and speed.



Constraints

1. Most services are myopic in nature, (e.g, watching utilization to flag for increasing capacity). The upstream change driving volatility in the service is unknown.
2. Likewise, downstream dependencies are notified when the service disrupts them. They in turn ask to be forewarned, but that information is still unknown.
3. How all these discrete services and upstream and downstream dependencies work is not effectively captured anywhere. Documenting them is even less practical when they are rapidly changing.
4. When constraints are hit, it requires some diligence to discover if the reasons are worth the investment to address. However, if you are using cloud services, you can automate capacity increases. Does anyone know the business reasons why that was done? Was it recorded anywhere?



Steps

1. Implement machine learning services or connect selected SaaS provider(s) respective services (defined here as an "engine").
2. Connect the engine to analytical models and gathered historical data.
3. Begin training the engine and modelling known good and bad conditions, improving accuracy using known historical information.
4. Compare the engine's model to the current model and dashboard and start testing predictions. If the predictions are wrong, there is more learning (or potentially other data) needed.
5. Move from binary decisions to detecting complex behaviors, to regression predictions about optimal pricing, or likely costs, etc.
6. Start applying this learning (predictive algorithms) more broadly to meaningful situations and strategic planning.



Forces

- As business processes the shift to digital and IT services are virtualized, friction is being removed. At the time when it took too long to procure more capacity, friction was a bad thing. Today, and in the very near future, that may not be the case.
- One lesson distributed technology has taught, is that technology breeds more technology. More of this means more of that, which in turn requires more of something else (which uses the first thing) and the cycle continues. A self-fulfilling prophecy.
- Add these two concepts together and take into account that business and technology are combining.



Results

- Models can be continually refreshed with updated dependencies improving current situational awareness and understanding.
- Viewpoints on business activity, outages (failure predictions), utilization (waste), behaviors (fraud, etc.) and risk (operations heat map) can be built and continually refreshed.
- A digital dashboard and trends analysis to report on – which is mostly automated and less error-prone.
- Platform to evolve and incorporate external business events and activity with an eye toward digital business models and direction.
- Direction is really only limited by data.
- All aspects of the business have become digital, completing the digital platform.



Reference View

