Government Conceptual Enterprise Architecture

A logical high-level enterprise architecture for government use cases.

Authors

Darren W Pulsipher

Anna Scott

Overview

A cloud or edge-only architecture can solve a few of today's data problems. Resources from edge to cloud are often needed to optimize the solution for effectiveness (e.g., real-time) and cost (e.g., total TCO). Here, we present a conceptual architecture that targets multi-hybrid cloud and edge computing strategies where data and applications can be moved as needed to optimize results and deliver on the mission.

To enable a future-proof and expandable system, it is essential to understand how different parts of the system relate to each other and establish isolation layers (through standard interfaces or abstractions). This isolation allows the various systems in the solution to "grow" in parallel with minimal effect on each other. It also allows multiple vendors to be used within each layer, making the system flexible and adaptable with no vendor lock. Establishing standard interfaces between the sub-systems further allows the easy adoption of new features for hardware or software based on use case needs.

The logical enterprise architecture (Figure 1) shows how the different subsystems (layers) fit together. We will first briefly overview each layer and then cover each in depth.

Table of Contents

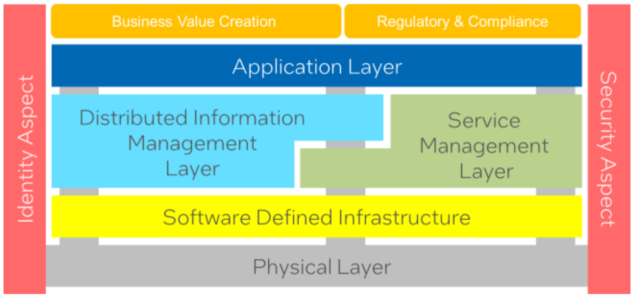


Figure 1 High-level Architecture

- Application Layer –Development, testing, and deployment of applications in the solution space. This is the primary interface to Business Value creation and regulatory and compliance requirements.
- Distributed Information Management Layer – Management (curation, governance, lifecycle management, and tagging) of data across a heterogeneous infrastructure (Cloud, Data Center, Edge, and Client).
- Service Management Layer – Deployment, monitoring, and provisioning services (containers) in the solution.
- Software Defined Infrastructure – Responsible for the solution's management (deploying, monitoring, and provisioning) of the Physical Layer.
- Physical Layer – Responsible for the command, control, and monitoring of the solution's physical devices (Compute, Storage, Network, and Accelerators).
- Security Aspect – Gives a standard security model across the subsystems of the solution.
- Identity Aspect – Gives the ability to uniquely identify and attest the identity of users, hardware, applications, services, and virtual resources.

Many organizations already have many of these subsystems in their toolbox of solutions, so developing a solution from scratch is rarely necessary. The recommended starting point is to understand what you are currently using, how those tools fit together, and how they interface. It is often possible to build in state-of-the-art features (use of AI models) by expanding current capabilities. No one wants to rip and replace systems in place and working. Instead, the goal is to utilize the current devices as a foundation to build upon for the future end goal. A roadmap of technology and process changes shows how the foundation can be built upon to achieve long-term architecture.

Application Management Layer

The Application Management Layer (AML) manages applications and workflows and the development, testing, deployment, and updates of those applications and workloads.

The AML contains abstractions that help App Development, DevOps, and IT Operations manage complex workflows and applications through the application development lifecycle. See Figure 2. The AML sits at the top of the system stack and communicates directly with the Distributed Information Management Layer and the Service Management Layer. It also leverages the Identity Aspect and Security Aspect layers. Because certain applications operate most efficiently on specific HW types, the Physical Layer and its capabilities are exposed to the AML. For example, training a large language model runs on Intel Habana hardware most efficiently.

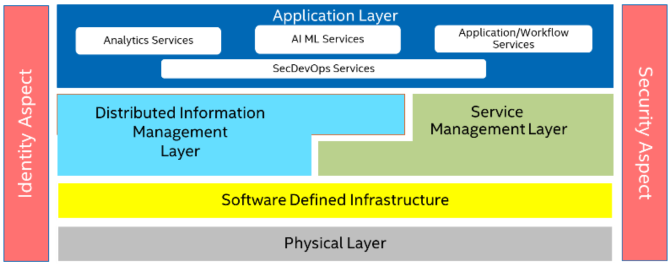


Figure 2 Application Layer

The AML contains sub-packages that group common-off-the-shelf tools together. In the Analytics Services package, standard tools for data analytics can be found, including business data understanding, modeling, and simulation. AI/ML Services represent various tools and services focusing on Artificial Intelligence and Machine Learning algorithms and solutions. These two packages take advantage of the application and workflow services that allow these solutions to be orchestrated at the highest level of integration by providing a standard definition framework to show how these applications and workflow interact.

Several tool suites have been built to aid DevSecOps. These tools are grouped in the DevSecOps Services and include Automation Frameworks (Salt, Chef, puppet, ansible), CICD tools (Gitlab, GitHub, Jenkins, etc.), and Environment Management.

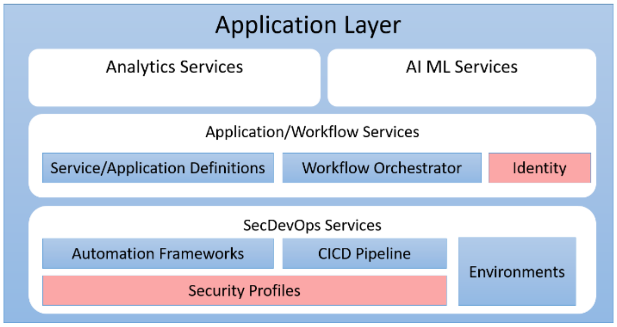


Figure 3 Application Layer Details

Physical Layer

The Physical Layer (PL) contains abstractions allowing better management across an ecosystem inside the data center, cloud, and edge devices. These abstractions give the ability to manage highly variable hardware configurations by describing the common operating and taxonomy of the devices. This architectural layer has the goal of addressing the following characteristics:

- Common Taxonomy
- Portability and Interoperability
- Security and Root of Trust
- Common Management Control Plane
- Performance Optimization
- Stability and Reliability
- Flexibility and Agility

The PL sits at the bottom of the Architectural stack but interacts with all other layers and aspects. It relies on Security and Identifying aspects to establish the hardware root of trust, identity, and data encryption at the lowest levels. Figure 4 shows the abstract layers established across Edge, Legacy, Data Center, and Cloud physical resources.

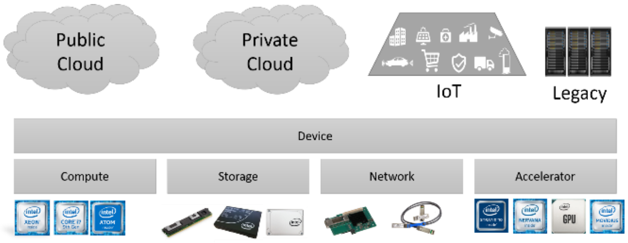


Figure 4 Physical Layer Abstraction

The critical element in this layer is called a Device. A Device contains one or more hardware elements, including processors, memory, accelerators, storage, and network capabilities. Each Device has a snapshot of its capabilities, hardware, and currently available resources in a "Profile" abstraction. See Figure 5 Device Abstraction The Device has a simple interface for control and telemetry through the Device Profile, providing the ability for the Software-Defined infrastructure layer (SDI) to deploy and provision applications and services to take advantage of the Device's specialized hardware.

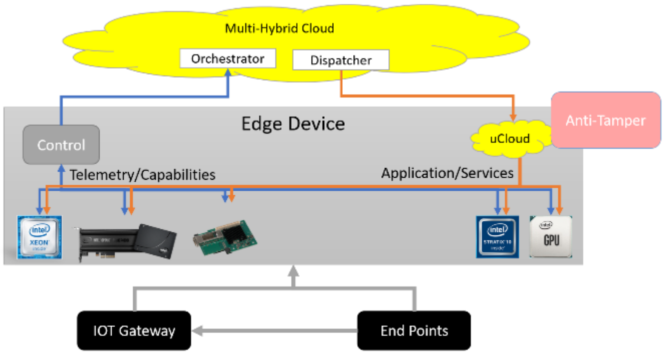


Figure 5 Device Abstraction

With the explosion of edge computing and sensors, the complexity of managing the devices in conjunction with the cloud and the data center has dramatically expanded. Managing 10s to 10,000s of devices is overwhelming for IT operations engineers, and many management and control architectures cannot scale appropriately. For this reason, the enterprise architect has created the Aggregated Device that allows the grouping of devices into collections that can be managed and controlled more easily. Aggregated devices can contain devices or other aggregated devices, giving the ability to have infinite layers in the hierarchy of devices, as shown in Figure 6 Device Aggregation.

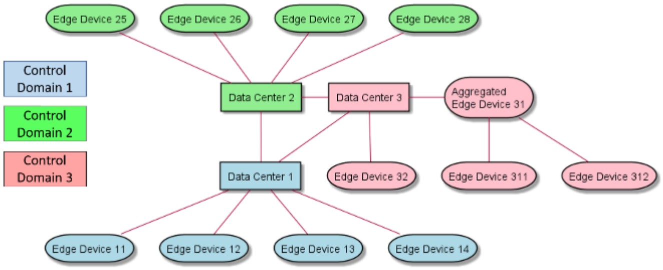


Figure 6 Device Aggregation

Often, organizations combine the physical management and the logical management of devices. Combining the physical and logical approaches is problematic as they create highly coupled, rigid, and fragile solutions that cannot adapt to change.

The enterprise architecture separates the physical and logical topologies, allowing for an architecture flexible to business and operating environments. Additionally, the two topologies give the flexibility to establish a true edge to cloud architectures, including setting up a cloud that spans resources in prem data centers, public clouds, and edge devices. It allows scheduling and managing applications and services across traditional boundaries.

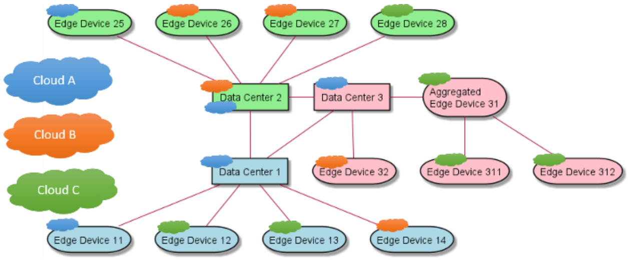
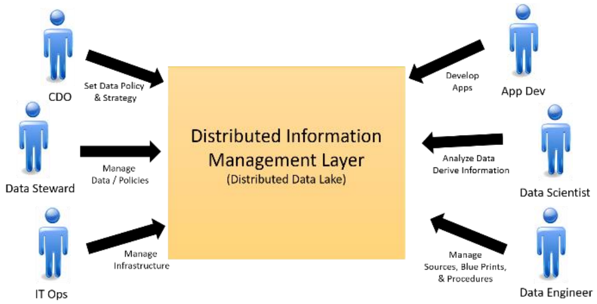


Figure 7 Physical and Logical Topologies

Figure 7 Physical and Logical Topologies, shows three clouds that share devices and span the control topology established for optimized IT operations. This flexibility allows clouds (logical devices) to adapt to changing environments. These changes can include everything from cyber threats, physical disasters, partial connectivity of edge devices, or even someone tripping over a network connection in the data center.

Distributed Information Management Layer

The Distributed Information Management Layer (DIML) manages data across the ecosystem, including data lifecycle management, data security and governance, storage infrastructure, analytics, data sources, and application data usage. Almost all architecture actors have some input into the DIML, including Chief Data Officer, Data Steward, IT Operations, Application Developer, Data Scientist, and Data Engineer.


CDO
Set Data Policy & Strategy
Data Steward
Manage Data / Policies
IT Ops
Manage Infrastructure
Distributed Information Management Layer
(Distributed Data Lake)
Develop Apps
App Dev
Analyze Data Derive Information
Data Scientist
Manage Sources, Blue Prints, & Procedures
Data Engineer

The DIML is the newest architectural element in Enterprise Architecture and is crucial in providing the flexible architecture needed for today's complex and demanding computing environments. It was developed to handle the complexity of managing data across multiple data centers, clouds, and edge devices.

The DIML subsystem supports multiple data architectures at the same time. This allows the Data Engineer to quickly build repeatable Blueprints geared to the different data architectural approaches based on the most efficient for a specific problem. As a result, the same system or solution can utilize various operating data models, including centralized architectures like Data Warehouse and Data Lake, and distributed architectures like Data Mesh, Data Exchange, Data Fabric, Data Mart, and Data Streams.

Evaluation of these data operating models has found three standard components across all data architectures. First, a Data Definition Framework can define data pipelines, categorization of data, and their generating or storage sources. Second is the Common data services like ingress, transform, store, and egress. The third is the Data Management component, which orchestrates the ecosystem's data movement, lifecycle management, and data governance. Figure 8 contains an overview of the DIML layer.

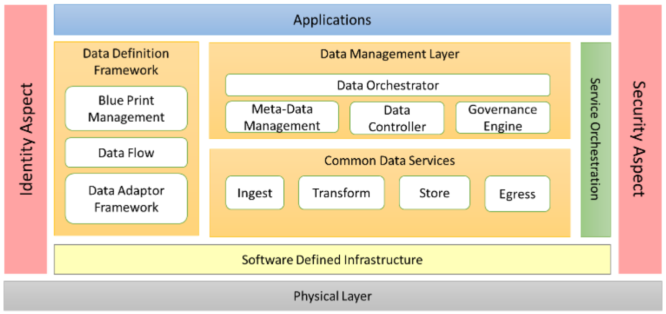


Figure 8 DIML High-Level Architecture

All of these approaches can be boiled down to two significant models of operation: centralized processing and distributed processing. The centralized processing approach utilizes data architecture to benefit the end users. There are many data architectures for centralized processing, two of which are Data Warehouses and Data Lakes. The centralized processing paradigm is good for some data use cases but not all. Many centralized data architectures fall apart as more systems move outside the traditional data center walls. This is where distributed processing architectures like Data Mesh become important. The architecture is flexible enough to handle the different modes of data processing.

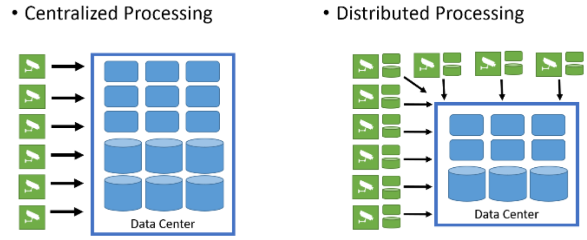


Figure 9 Centralize and Distributed Data

Data Mesh

In the Data Mesh architecture, moving applications close to the data or the data close to the applications is possible. Data processing is done on edge devices, and results are pushed to the data center/cloud to be linked. This contrasts with traditional Data Warehouses and Data Lakes, where data is stored in a centralized location.

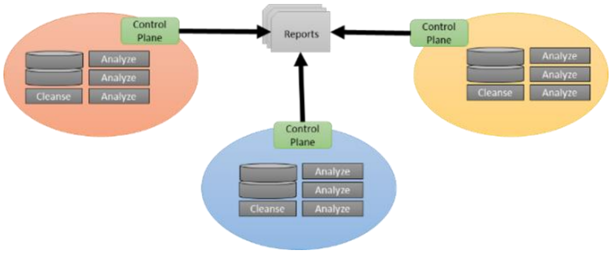


Figure 10 Data Mesh Architecture

Data Exchange

Another mode of operation is Data Exchange. This takes the Data Mesh concept and extends it to different classifications or owners of data. This mode limits the movement of data and who has access to it, making it ideal for Government and Healthcare, where privacy and classification regulations restrict data access. Data Exchange architecture allows policy gates to limit the data that can be passed back to the application requesting the data. It also allows running analytics/services in the geo-fenced data site.

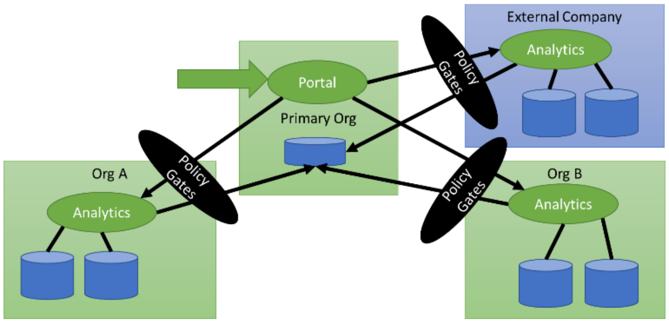


Figure 11 Data Exchange Architecture

Data Fabric

Data Fabrics are the natural next architectural evolution to emerge since they resolve some of the problems with Data Lake architectures caused by centralizing all of the data. Data Fabrics process data on the edge where the source generates the data. This distributed architecture follows much of the same path that cloud technology did in the early 2000s and includes centralized control, orchestration, and management of the data.

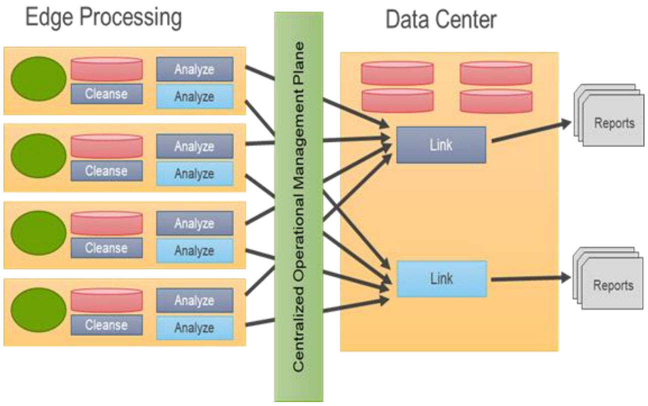


Figure 12 Data Fabric Architectures

Data Mart

A Data Mart is a small data repository for structured data specific to a department. Tailored to the detailed problem statements, they contain copie/s of data from a limited number of sources and typically a smaller data set. Data Marts usually limit access to the data and report to one organization or a small group of users in one organization. Data scientists leverage Data Marts to build complex analytical models, generate timely periodic reports that require highly predictable performance, and work with sensitive data and resulting reports.

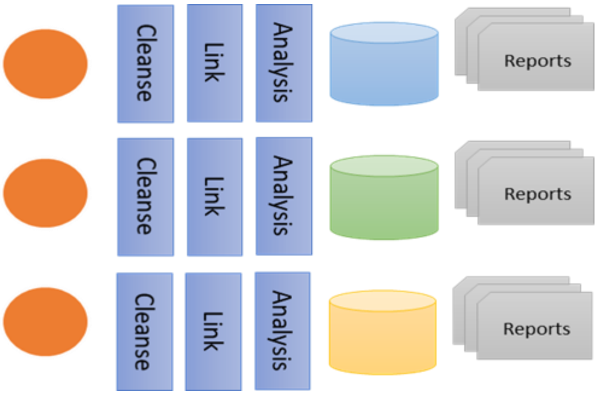


Figure 13 Data Mart Architecture

Data Streams

Data Stream architectures allow for processing data analytics in the stream of data. Each Data Stream manipulates the data as it is ingested and egressed to another application, report, or data stream. Data Streams allow for data to be used anywhere in the ecosystem, including on the edge devices, in the data center, in the cloud, and even in transit between the different types of infrastructure. Analytical reports can be generated parallel across multiple devices by combining data transforms through data streams. See Figure 14.

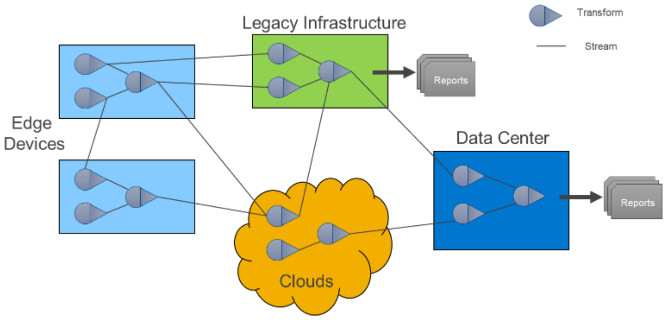


Figure 14 Dynamic Intelligent Data Streams

Service Management Layer

The Service Management Layer (SML) subsystem manages services, stacks, environments, and multi-clouds. The SML is a middleware layer in the architecture responsible for orchestrating and managing services across multiple clouds (public and private) and edge. The SML coordinates with the Distributed Information Management Layer and the Software Defined Infrastructure. It takes requests from the Application Management Layer to deploy services that make up applications and workflows.

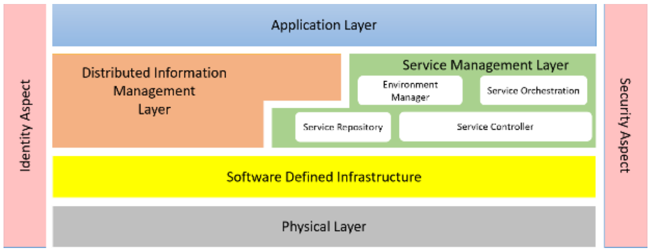


Figure 15 Service Management Layer

The main goal of the SML is to provide the Application Developer with a simple, repeatable, robust mechanism to deploy services into the multi-cloud and edge ecosystem. It must also offer IT Operations mechanisms to enforce cost, reliability, and security policies. Applications and Services are deployed to cloud assets based on these policies and can be run across cloud/edge boundaries as dictated by the IT policies enforced in the system. All communications between services should follow secure communication protocols as the IT policies dictate. The key is that a single portal or gateway should be used such that applications and services are deployed and managed automatically without human interaction. Decisions about where services should land should be automatic based on the IT policies established, not determined by the Application Developer or IT Operations Engineer.

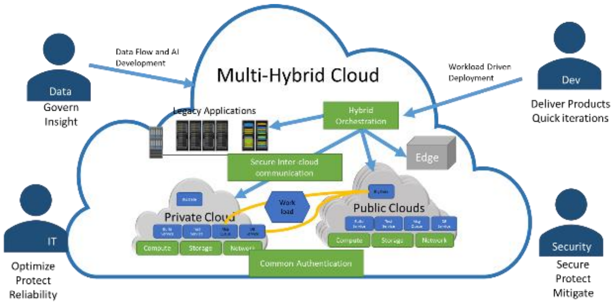
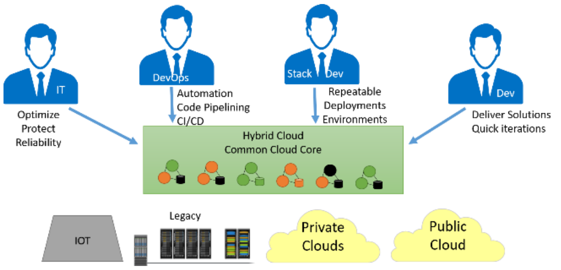


Figure 16 Multi-Hybrid Cloud Overview

The SML has several actors that work with the sub-system. Each one of these has a different motive for using the system. Even though some of their uses of the methods are the same, the reason they are using the system is very different.

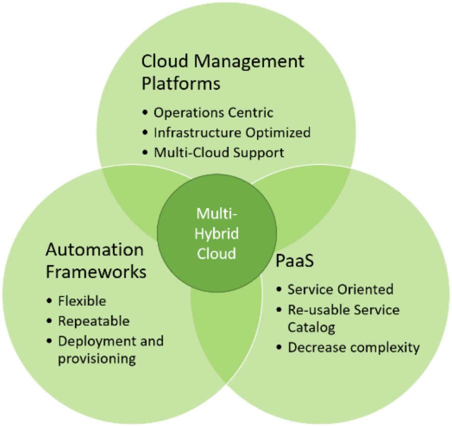
- IT Operations - Motivators include optimizing infrastructure for cost, protecting infrastructure and IP, and increasing reliability and resiliency.
- DevOps Engineer - Motivators include automating everything, streamlining code pipelining, and managing build and deployment with CI/CD.
- Application Developer - Motivators include repeatable and reusable service stacks, deploying services across cloud and environments.
- Stack Developer - Motivators include delivering solutions in quick iterations, concise break, fix, and deploy cycles.



The SML spaces are full of tools that can be integrated to deliver the use cases demanded by the actors in this space. The devices can be categorized into three major categories:

- Cloud Management Platforms - built for IT Operations focusing on multi-cloud support and management of infrastructure profiles across multi-clouds.
- Automation Frameworks - built for DevOps and Stack Development, focusing on providing and deploying software in a repeatable/reusable manner.
- Platform As A Service - built for Application Developers focusing on reusing services and decreasing the complexity of using those services to build applications.
- Edge?

These tool sets have been developed by and for specific actors. And the integration of these tools together helps to cover gaps that exist in the individual devices.



The SML has sub-systems as part of the architecture, including:

- Environment Management - manages environments (dev, test, prod) across multiple clouds
- Service Orchestrator - orchestrates the services in different environments.
- Service Registry – provides a centralized repository of service definitions.
- Data Coordinator - works with the DIML to orchestrate data and services.
- Security - works with the security policies and tools to ensure applications and services communicate securely.
- Provision Engineer - provisions software stacks and services on infrastructure
- Cloud Broker - manages the clouds (which cloud can manage which request)

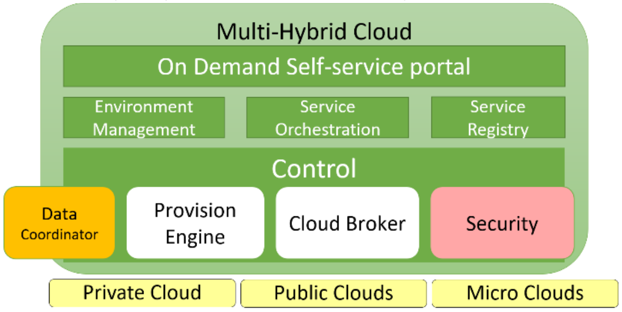


Figure 17 Multi-Hybrid Cloud High-Level Architecture

Software Defined Infrastructure

Software Defined Infrastructure (SDI) contains the abstractions for private and public clouds. The SDI layer is a familiar standard interface for all cloud resources-virtual, container, and bare metal.

The Software Defined Infrastructure Layer (SDI) is a middleware layer in the architecture. And fits between the Common Physical and the Distributed Information Management and Service Management layers.

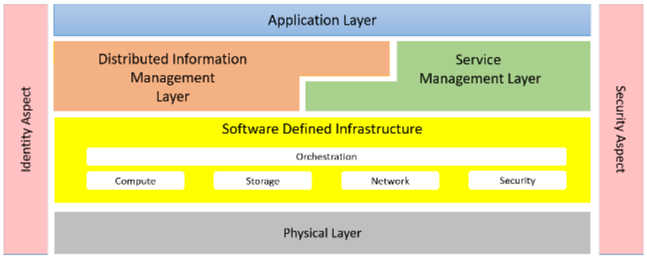


Figure 18 Software Defined Infrastructure

It is primarily responsible for IaaS operations and management. SDI architectural elements are well-known and established in the industry, with commercial and open-source product offerings available (VMWare, OpenStack, Nutanix, etc.). The critical elements of an SDI layer are Orchestration and Control, Infrastructure elements (Storage, Network, Compute/Accelerators, and Security), and a Common Infrastructure API Gateway.

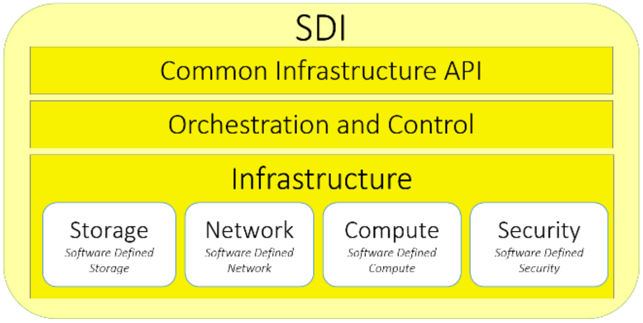


Figure 19 SDI Internals

These key architectural elements are minimal viable features for a standard interface to IaaS solutions to be used in a Physical Layer. The ability to interact with a common API interface regardless of the type of Cloud is essential for interoperability between the private and public cloud offerings. To include Edge Devices into the ecosystem, the concept of a micro cloud was developed with the same minimal Common Infrastructure API. This concept extends the boundaries of the cloud to the edge and gives the ability to manage infrastructure and applications across a traditionally tricky border.

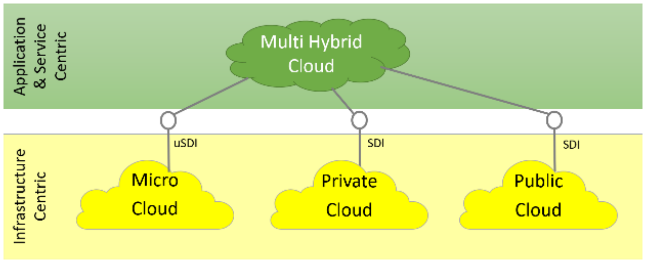


Figure 20 Cloud Abstraction

The shared infrastructure API allows Multi-Cloud Orchestrator from the Service Management Layer to request infrastructure (Bare metal, Virtual, or containerized) to deploy complex applications across several cloud offerings.

Identity Aspect

The Identity Aspect provides identity through all layers of the Enterprise Architecture. This aspect is responsible for the trusted identity of users, devices in the data center, the cloud, the edge, services and applications, and data. Having a standard identity management system is critical to having consistency in the system. This identity must be trusted so that the data, applications, and hardware can deliver solutions that can be used confidently.

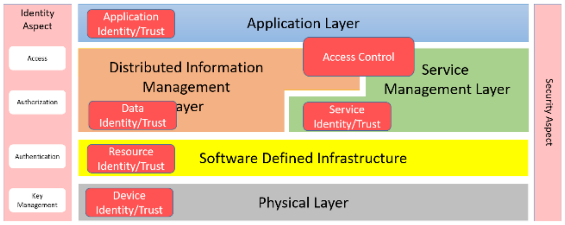


Figure 21 Identity Aspect Mappings

The Identity Aspect has critical sub-systems that help manage identity: Access, Authorization, Authentication, and Key Management. Authenticating an entity in the system is the first step in identity management. Keys are used to certify and attest to the authentication of an entity, human or machine. Once an entity is authenticated, it is given authority to access other resources in the system. By providing identity to every element in the design, mishaps in security can be mitigated and better controlled. Given the new focus on Zero Trust Architectures, we constantly re-verify identity, and authorization is revoked if credentials do not match those expectations.

Security Aspect

Security Aspect contains security tools and subsystems that are used throughout the architecture. The Security Aspect is used in every layer of the architecture. The aspect was developed so that all layers in the architecture have a common mechanism to perform common security use cases, such as encryption, detection, remediation, and root of trust.

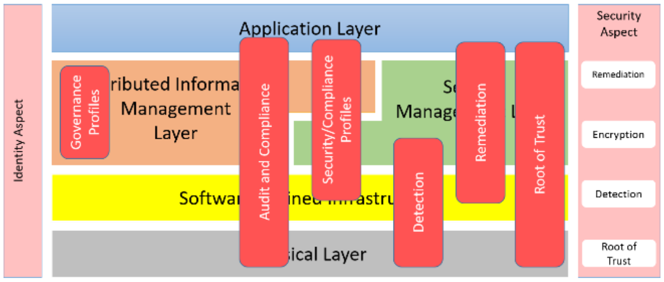


Figure 22 Security Aspect Mappings

Conclusion

With the complexity and ever-changing ecosystem of technology, business process innovation, and operating environments, IT professionals and architects can't keep up with the trends. New tools are often organically adopted and deployed without understanding their interactions. This can lead to rigidity, vulnerabilities, and unreliability in the solution ecosystem. This Enterprise Architecture attempts to show different subsystems and how they interact. It provides IT architects with a map to guide them through the deployment minefields of modern solution deployment and management in this complex ecosystem.

[1] All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest Intel product specifications and roadmaps.
Intel technology's features and benefits depend on system configuration and may require enabled hardware, software, or service activation. Performance varies depending on system configuration. No computer system can be secure.