Reference Architecture Models for Big Data and Analytics

by Arun Viswanathan, Principal Solution Architect

When a data platform is created without guidance or planning, best practices and finer points during the data preparation stage are either ignored or overlooked, resulting in multiple challenges at later stages. After the data platform goes live, adopting a retroactive approach to fix errors and close gaps is expensive, and the challenges can be formidable and difficult to overcome. Big Data architecture and technologies, which are readily available and accessible, typically focus only on the technologies in the core data platform. Data architecture needs to account for cross-cutting capabilities, non-functional requirements, and data governance. This white paper describes the reference architecture for Big Data and Analytics and a checklist of components you can consider and evaluate when architecting an enterprise data platform.

Context and Problem

Big Data Architectures are well established, and the supporting Big Data technologies are ubiquitous and continue to evolve. While these architectures and technologies are readily available and accessible, it is crucial to have a guiding plan to design a full-thought data platform architecture. Once the data platform is built, it becomes extremely costly and painful to go back and correct the missed aspects. Hence, we need to ensure that some of the finer points of building enterprise data capabilities are also thought through and addressed while designing the data platform architectures.

This white paper shares our perspective on the data platform architecture based on GlobalLogic's experiences with different customers. It provides a checklist of the elements covering not only core logical components of a data platform like ingestion interfaces, data processing, data storage, etc., but also other areas such as Non-Functional Requirements, observability, metadata management, data retention, data quality, and data governance. However, this document does not intend to present technology selection, design, or implementation-level guidance.

Taxonomy

This section describes the taxonomies referenced later in this white paper.

Terms	**Description**
Systems of Record (SOR)	The transactional data stores of Feature Systems.
Systems of Truth (SOT)	The aggregated data stores of a Data Platform. The primary purpose of an SOT is to provide an aggregated and cross-enterprise view (or a 360-degree view) of critical enterprise data entities.
Identity and access management (IAM)	A framework of policies and technologies that enable authoritative inventorying and authentication capabilities for human and system identities.
Role Based Access Control (RBAC)	An approach defined around roles and privileges that restrict system access to authorized users and allows users to perform operations based on the permissions assigned to the user’s role.
Data Lifecycle Management (DLM)	Data Retention concerning DLM generally focuses on data from a container perspective rather than the actual contents of the data. For example, a DLM policy could be to delete trashed emails older than one year permanently.
Information Lifecycle Management (ILM)	Data Retention concerning ILM generally focuses on data from a content perspective and less on the surrounding container within which the data may be held. For example, an ILM policy could be that Personal Information can only be kept for as long as Services are being provided.

Scoping the Data Platform Within the Enterprise

Every organization should maintain an enterprise city map that inventories all notable applications, services, and platforms across the enterprise. The scope of systems covered within this City Map could be limited to only those that manifest or support customer-facing or partner-supporting capabilities. Other systems, such as ERP and HR, may not be in the scope of the Enterprise City Map unless they have an integration interface with customer-facing or partner-supporting systems. In that case, the integration interface could be in the scope of the Enterprise City Map.

Figure 1 shows a conceptual enterprise ecosystem consisting of different business unit systems, shared components, domain services, shared platforms, platform services, and hosting and infrastructure services.

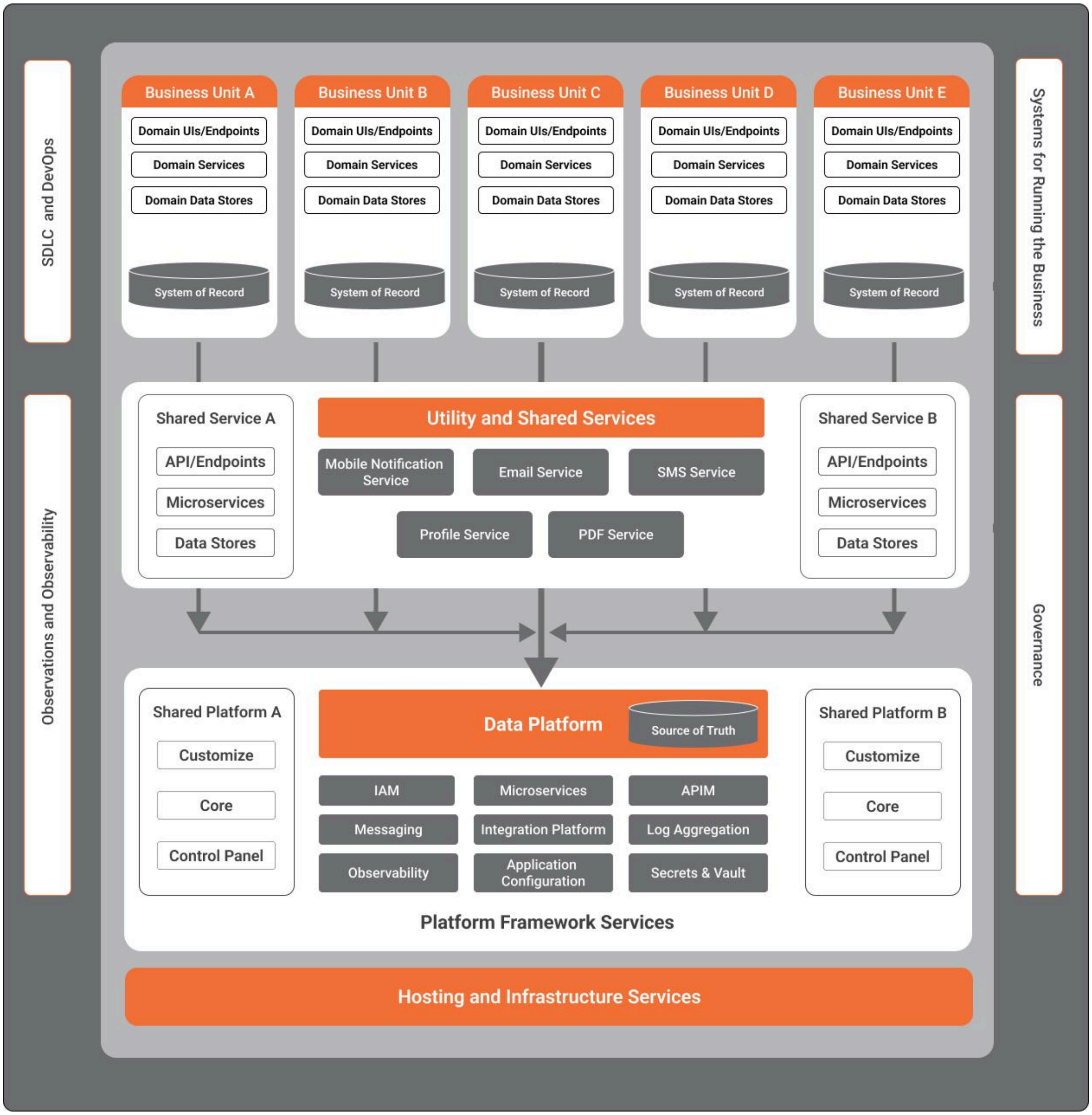

Figure 1 - Enterprise City Map (Source)

Business Unit Systems

Each business unit has applications that provide a set of domain-specific functionality either to end-users or other systems. For example, a financial system could provide end-user UI application capability or data access endpoints to Partner Systems.

Shared Services

Shared Services are domain services that have been factored out of the higher-level Feature Systems. Shared Services are typically manifested as services with associated compute and storage aspects. For example, a central shared services team can maintain shared implementations and deployments of common capabilities like notification service, email service, download service, etc.

Shared Platform

Shared Platforms provide lower-level capabilities upon which higher-level services are built. Platforms offer core capabilities that do not contain domain-level aspects, along with a customization model to implement domain-level functionality. For example, a Microservices Platform could offer container, clustering, and chassis capabilities as a shared platform, which can then be used to build Domain Microservices.

There are two approaches to implementing Shared Platforms:

- **Central Deployment** - The Platform is deployed and managed by a central team within the Enterprise. The various feature teams onboard it with their customizations. For example, a central team within the enterprise deploys and manages a shared instance of Kafka for notable/cross-enterprise messaging capabilities, and then various feature teams "on-board" to the centralized messaging platform with their various topic needs.

- **Distributed Deployment** - The Platform is deployed and managed by the various feature teams as needed. In this model, standardization of technology, versions, and dependent libraries provides a consistent approach for low-level platform interfaces. For example, a central team establishes the Confluent distribution of Kafka (with a particular version) as the Enterprise Standard, and the central team also implements a shared messaging library that teams can then use for consistent/controlled interfacing. In the distributed deployment scenario, teams in business units may choose to deploy/maintain their own instance of the standardized technologies and shared libraries.

Figure 1 showed how the different business unit systems store the transaction data or systems of record (SOR) within the respective systems. These data types are all ingested into a centralized data platform through the different ingestion adapters and stored in the data lake or data warehouse within the platform. The data platform functions as the system of truth (SOT) for data across the enterprise.

Note that the data platform is one of many other platforms, such as microservices, integration, messaging, etc., present within the platform services. All of these platforms advantageously share common utility services and frameworks.

Enterprise Data Platform Reference Architecture

Within the enterprise city map, an enterprise data platform plays an important role in bringing together the data from different systems and sources into a unified platform where data can be analyzed for deriving insights. It is important to note that we did not invent data platform architectures; instead, many such reference architectures are available in the industry. Several cloud vendors provide reference architectures that leverage services specific to the cloud vendor. Generic industry architectures, such as Lambda and Kappa, can be leveraged based on specific data processing design patterns.

Figure 2 summarizes five reference architectures:

- Lambda architecture
- Kappa architecture
- Google Cloud Platform Analytics Lakehouse
- AWS Modern Data Analytics
- Azure Analytics End-to-End

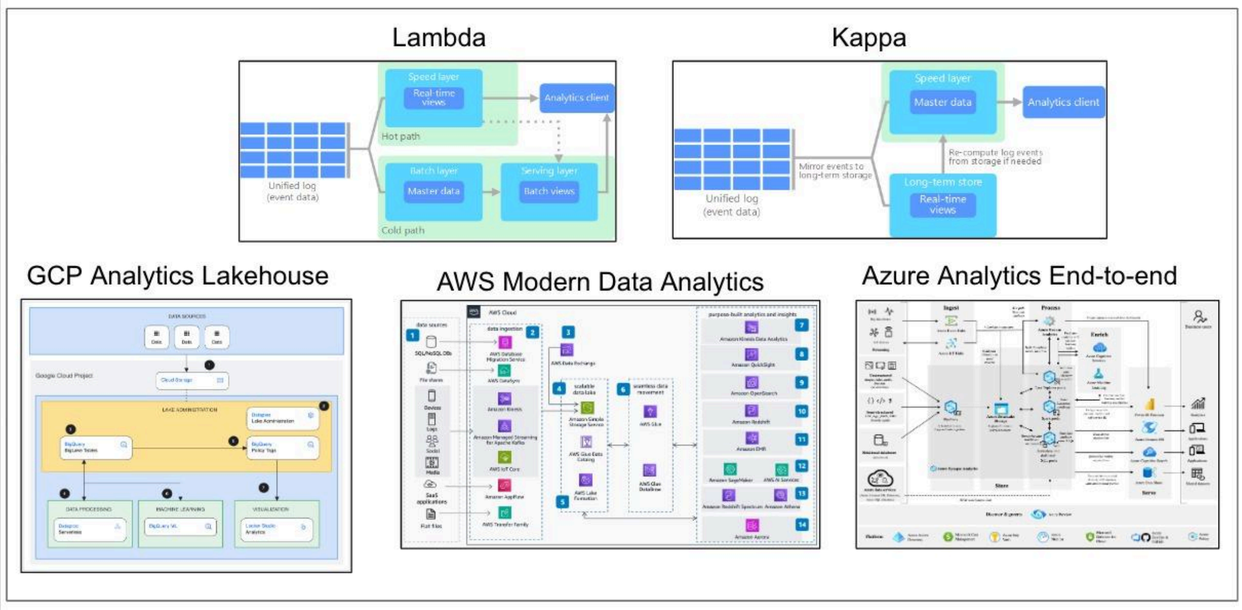

Figure 2 - Reference Data Platform Architectures

At GlobalLogic, we are data practitioners with experience working with different customers and have taken the liberty to put together a data platform reference architecture based on our experiences.

In Figure 3, we provide our perspective on a data platform reference architecture, which aligns with the widely accepted view of a data platform. The reference architecture includes the functional component elements rather than technology or vendor-specific services so that it can be applied across different cloud vendors and data architecture requirements.

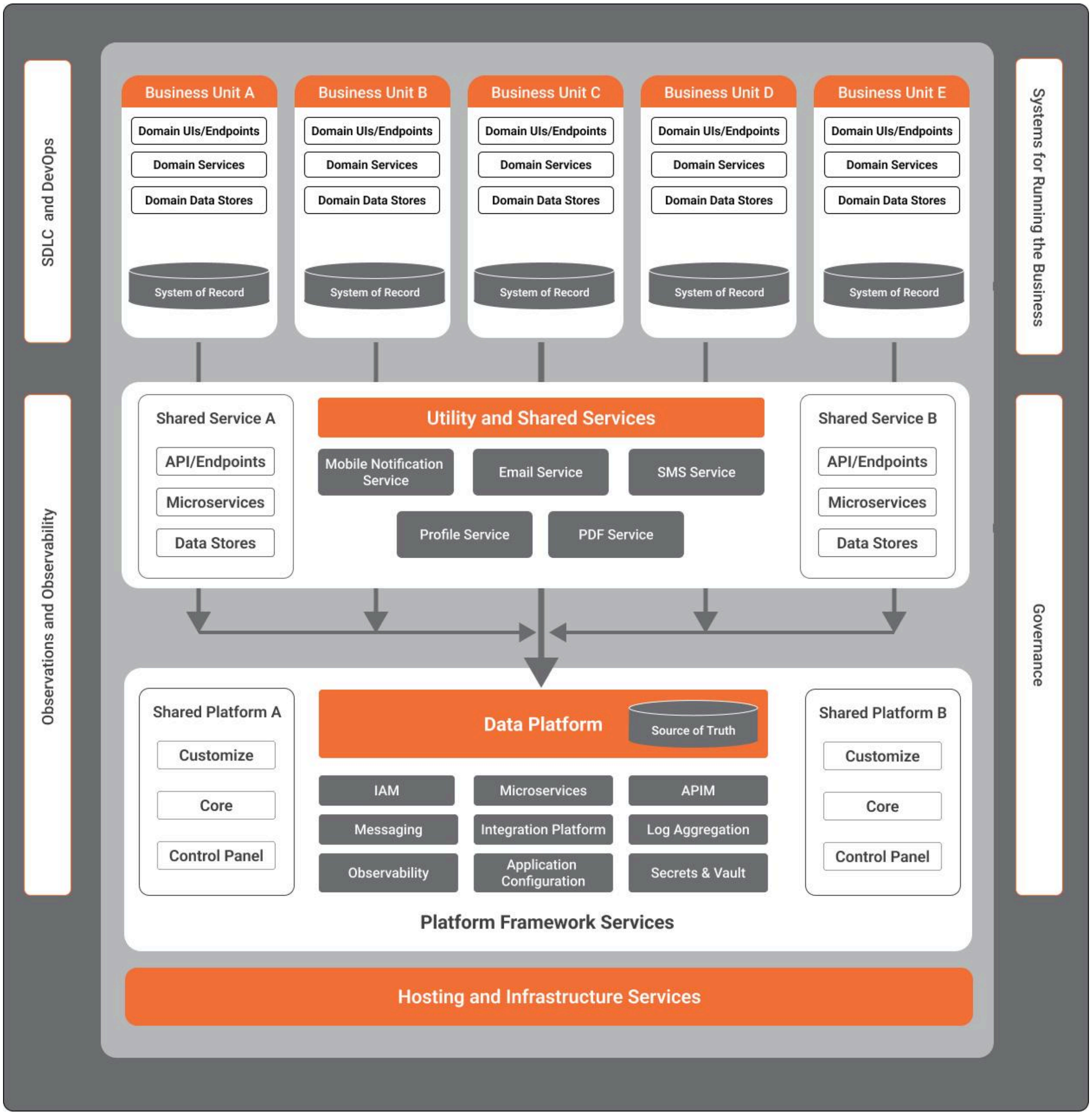

Figure 3 - Enterprise Data Platform (Source)

The following section will provide a walkthrough of the different component layers.

Components of the Data Platform

In this section, we provide a brief overview of the components in the reference architecture that are considered necessary for a complete and well-thought-out approach regarding enterprise data platforms.

Data Producers

Data Producers are the source systems from which the data must be integrated into the data platform. These producers can include internal transactional systems, internal sensor systems, internal business systems, client or partner systems, or external systems producing licensed / open source data.

Ingestion Interfaces

Depending on data velocity, you can use different ingestion interfaces to integrate with data source systems. The ingestion interfaces can vary from periodically scheduled batch ingestion, near real-time micro-batch ingestion, and real-time event streaming-based ingestion mechanisms.

In Batch ingestion, source system data is fully loaded into the target systems. In micro-batch ingestion, data is collected in small batches and processed. In streaming-based ingestion, data flows from real-time sources like sensors or events through event streaming platforms and is processed in real time. Middleware-based replication and synchronization using tools such as Debezium or GoldenGate offer data integration approaches that track all changes in a data source from databases or data warehouses and replicate the changes to the data platform in real-time.

Wrangling and Refining

The Medallion Architecture, which includes the Bronze, Silver, and Gold layers, is a common data integration approach used to wrangle and refine data as it is ingested into the Data Lake. The ingestion layer, or the Bronze layer, is where the ingestion interfaces first ingest data in raw format. A copy of the data is archived for future reference, validated, and stored in a standardized format in the refined or Silver layer. This data is then transformed, enriched, and curated before being stored in the Gold layer that serves business needs such as BI reporting, data science, and ML.

Data Lake

The different layers of the medallion architecture are stored across the data lake and data warehouse as applicable. The data lake can consist of polyglot storage systems, including object/file storage systems, relational databases, NoSQL databases (like document databases, graph databases, key-value stores, and columnar databases), or search systems. Depending on the usage, data type, and data volumes, one or more of these storage systems can constitute the data lake in an enterprise. Alternate architectures, such as a data lakehouse, combine the capabilities of a data lake and a data warehouse. Data mesh and data fabric are other data architectures that are gaining popularity and traction among enterprises that need a distributed data architecture.

Systems Of Truth and Governance

This group of components includes capabilities around Data Governance and acts as the source of truth, such as data catalog, data quality tools, master data management, and data warehouses. The data catalog capability curates the data stored across the platform with required metadata to make it easily discoverable by consuming applications. Data catalog tools can additionally capture data lineage to help trace the impact of schema changes downstream in the pipeline.

The data quality capability performs validation and quality checks on data being brought into the platform. A related capability is data observability, which builds on the concept of data quality and brings together some capabilities from the data governance and control plane to proactively track the overall health of an organization's data systems and data delivery mechanisms. Another important capability for maintaining the SOT is a master data management tool that helps store and manage *golden records* within the platform. Some enterprises also use data warehouses and data marts for storing modeled and aggregated data. Data governance also includes other capabilities, such as governing data security, setting data retention policies, and managing and enforcing data contracts for data providers and clients.

Access Interfaces

The data in the platform is only helpful if it is accessible to consuming applications that empower users to analyze the data and derive insights. Consuming applications need to be enabled by well-thought-out access interfaces that are essentially contracts with the rest of the enterprise ecosystem.

Onboarding and offboarding of identities and roles to and from access interfaces are essential to consider. It should be possible to formally onboard new users and roles and offboard existing users and roles from the platform.

Several approaches for onboarding consuming and partner systems into the data platform include:

- Data discovery through a catalog
- Industry-standard polyglot data serving APIs
- API developer portals
- Custom push/pull sync/async APIs
- Push mechanisms to send files through file delivery protocols like SFTP
- Onboard data consumers to hosted BI/analytics notebooks or enable them to bring their own tools
- Data virtualization
- Replication and notification interfaces
- Data sharing marketplaces

Depending on the security policies and user permissions, the consuming applications will be provided access to consume the curated data. Additional access control capabilities may need to be built into the data serving layer to protect data access according to the policies.

Control Plane

Managing a Data Platform needs a Control Plane for administering the platform and catering to cross-functional capabilities such as managing security through IAM and RBAC policies, data observability, metrics, monitoring, and job tracking of data pipelines and processing that occurs across the platform. It is essential to consider, implement, and apply different cross-cutting concerns such as centralized logging, authentication, authorization, configuration management, secrets management, application gateway, internationalization, localization, metrics, monitoring, traceability, etc, across the data platform.

Non-Functional Requirements

Non-Functional Requirements (NFRs) are another important aspect of the data platform. Some important aspects to consider are technical SLAs, such as system uptime percentage, average system response time, UI load time, average endpoint response time, scalability, high availability, disaster recovery (HA/DR), etc. Architects must also understand and qualify the process-related aspects such as SDLC-related requirements, DevOps-related strategy and automation, quality, security, and data compliance.

Data Consumers

Data Consumers are downstream systems that consume the data in the data platform through different means based on applicable permissions. These consumers can be internal enterprise systems, external Data as a Service (DaaS) connectors, BI/reporting tools, or data analytics programs. These consumers access the data through the defined access interfaces and the permissions granted to them.

Conclusion

This white paper offered various perspectives on Big Data and Analytics reference architectures to promote well-thought architecture approaches. We reviewed the enterprise city map concept, asserting that every enterprise should have one and include a data platform capability as a first-class citizen. This white paper also provided a view of data platform architecture based on practitioner experiences and existing industry standards. We finally covered a brief overview of the different component elements, including functional and non-functional requirements that need to be considered by architects while designing a data platform.

About GlobalLogic

GlobalLogic, a Hitachi Group Company, is a leader in digital product engineering. We help our clients design and build innovative products, platforms, and digital experiences for the modern world. By integrating our strategic design, complex engineering, and vertical industry expertise with Hitachi's Operating Technology and Information Technology capabilities, we help our clients imagine what's possible and accelerate their transition into tomorrow's digital businesses. Headquartered in Silicon Valley, GlobalLogic operates design studios and engineering centers around the world, extending our deep expertise to customers in the automotive, communications, financial services, healthcare & life sciences, media and entertainment, manufacturing, semiconductor, and technology industries.