

The logo icon for SMARTDIH, consisting of three vertical bars of different heights and colors: a yellow bar on the left, a blue bar in the middle, and a blue bar on the right.

# SMARTDIH

Product Overview

## Contents

1. Introduction to Smart DIH .....	5
1.1. What Is a Digital Integration Hub (DIH)? .....	5
1.2. Is DIH The Same as In-Memory Data Grid? .....	8
1.3. What's the Difference Between an Operational Data Store and a DIH? .....	9
1.4. What Is Smart DIH? .....	11
1.5. What Is a Space? .....	12
2. Data Integration Layer .....	13
2.1. Objectives .....	13
2.2. Data Pipelines .....	13
2.3. Pluggable Connectors .....	15
2.4. CDC Based Pipelines (Change Data Capture) .....	15
2.5. Addressing Changes in Data Structure .....	16
2.6. Data Validation and Cleansing .....	16
2.7. Data Registration .....	17
3. Hosting Layer .....	19
3.1. Objectives .....	19
3.2. Functionality .....	19
3.3. Data Modeling .....	20
3.6. Data Tiering .....	20
3.7. Data Exploration .....	22

3.8. Data Validity and Freshness Analysis .....	22
3.9. Performance Accelerators.....	23
3.10. Scaling .....	24
3.11. Backup and Persistence .....	24
4. Digitize.....	26
4.1. Objectives.....	26
4.2. Functionality.....	26
4.3. Data APIs.....	27
4.4. Low-Code Microservices Creation .....	28
4.6. Microservice Operationalization and Lifecycle .....	30
4.7. Event-Driven Microservices .....	31
5. Management & Control.....	32
5.1. Objectives.....	32
5.2. Infrastructure-Level Management .....	32
5.3. Application-Level Management .....	33
5.4. Change Management.....	34
6. Security.....	37
6.1. Data at Rest Security.....	37
6.2. Data in Transit Security (TLS).....	38
6.3. Access Audit.....	38
7. Summary .....	39

© GigaSpaces Technologies Ltd. and its affiliates ("GigaSpaces"). All Rights Reserved Worldwide.

Confidential and Proprietary Information of GigaSpaces. All materials (regardless of form and including, without limitation, software applications, documentation, and any other information relating to GigaSpaces, its products or services) are the exclusive property of GigaSpaces. By reviewing these materials, you agree to not disclose these materials to any third party unless expressly authorized by GigaSpaces, and to protect the materials as confidential and trade secret information. Any unauthorized review, retransmission, dissemination or other use of these materials is strictly prohibited. All materials found herein are provided "AS IS" and without warranty of any kind and GigaSpaces disclaims all liabilities of any kind relating to the materials provided herein and their use by any third party.

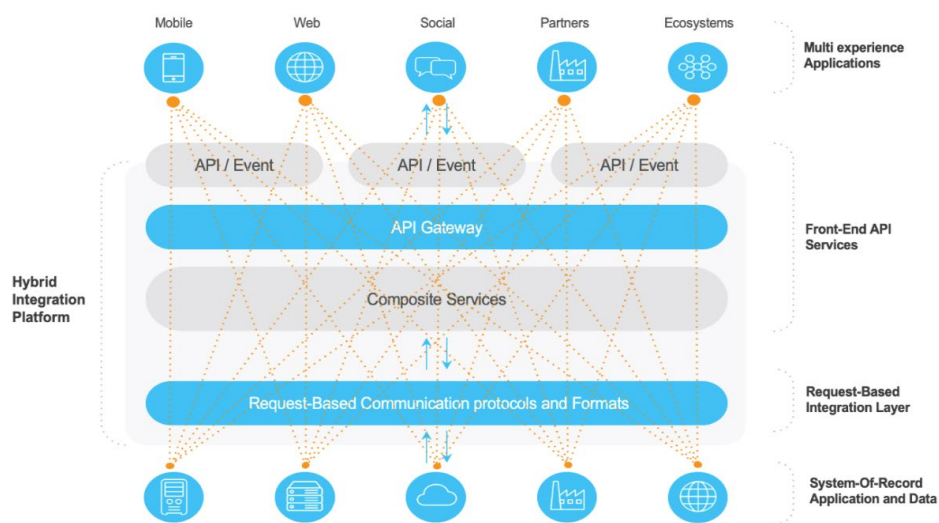
For detailed Terms of Use see <https://docs.gigaspace.com/terms-of-use.html>

# 1. Introduction to Smart DIH

## 1.1. What Is a Digital Integration Hub (DIH)?

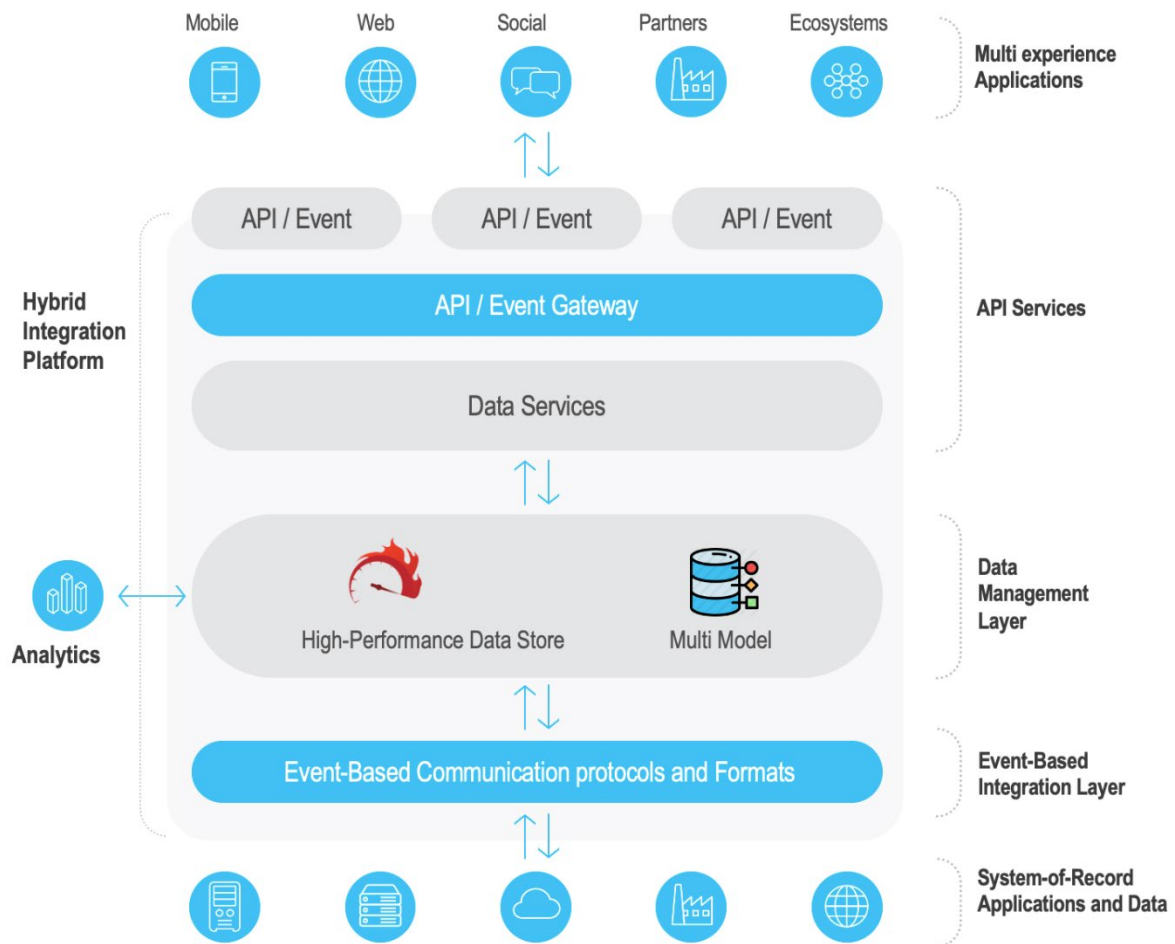
A Digital Integration Hub is a solution designed to power business acceleration. As organizations go through digital transformation, they need to introduce new digital channels and services, powered by data that originates in multiple systems of record (SoR).

Traditionally, digital applications access SoR data with a request-based integration. However, many legacy SoRs were not designed to support the speed, scale and volumes expected from digital channels. As a result, digital applications are not always available, users experience poor performance, and SoR application access fees increase exponentially as the volume of digital transactions grows. Most importantly, building new digital services becomes a lengthy and slow process, and enterprises struggle to meet the need of providing new and innovative digital services to their customers. This is due to the strong coupling between digital applications and the SoR.



*Conventional request-based integration architecture*

The Digital Integration Hub is a solution designed to address these challenges. It decouples the SoR from digital applications, allowing app developers to focus on business logic and innovative services, without worrying about integration with SoR applications.



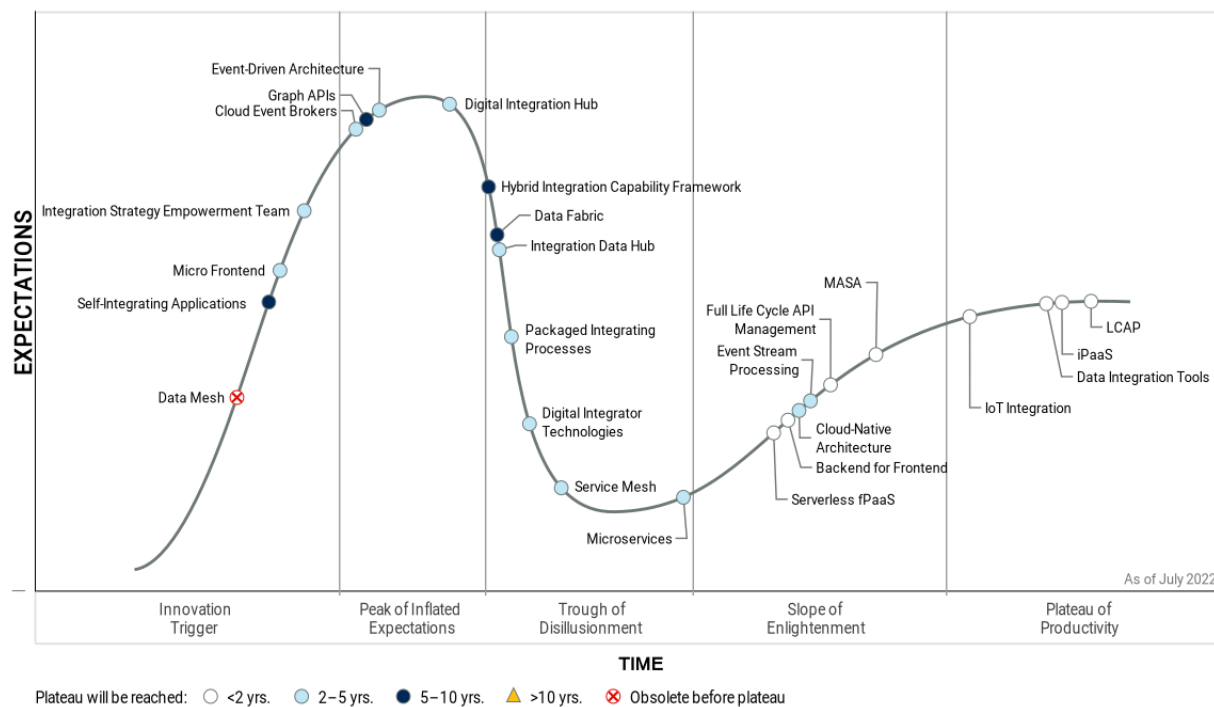
*A Digital Integration Hub*

The benefits of a DIH enabled event-based integration architecture include:

- **Business acceleration:** shortening time-to-market of a massive amount of new digital services

- **Enhanced Customer Experience:** guarantees low latency response times across digital channels, and high concurrency to meet SLAs
- **Always-on** digital applications, even when SoR are down or inaccessible
- **Developers can focus on innovation**, instead of managing and consolidating access to data
- **Systems of record** are protected from excessive workloads and bottlenecks
- **Reduced costs** are achieved by eliminating unnecessary access fees to SoR applications

The Digital Integration Hub concept is evangelized by Gartner. It is also featured on the Gartner Hype Cycle for Application Architecture & Integration, right at the top of the hype diagram. It is expected to reach the Plateau of productivity within 5-10 years.



*Gartner, Hype Cycle for Application Architecture & Integration 2022*



## 1.2. Is a DIH the Same as an In-Memory Data Grid?

In-memory data grids and other caching solutions such as key/value stores are commonly used to accelerate application performance and achieve low latency access to data. But while there are some overlapping functionalities between caching solutions and DIH, a Digital Integration Hub is very different from a cache. In fact, caching can be considered as one of several components that comprise a DIH solution, namely the high-performance data store at the heart of the DIH. However, it takes many more components and a lot of do-it-yourself system integration work to transform a cache into a full-blown DIH.

The main differences include:

	IMDG / CACHE	DIH
Objective	Accelerate performance and lower data access latency	Accelerate digital transformation and innovation
Scope of Applications	Typically, one known application	Massive number of digital applications and services, some of which are not designed or known upfront
Scope of Data Sources	Typically, one database	Typically, multiple databases and systems of record on multiple platforms
Scope of Data	Frequently used data to optimize performance	All relevant data to support digital channels
Importance	Usually, a tactical tool to address a specific bottleneck	Usually a strategic, cross-enterprise initiative
Users	Highly skilled developers	Data engineers, service developers, DevOps, Integrators, SOC engineers, system admins
Data Integration	Not included. Requires coding and external tools	Integration with databases and data stores



	IMDG / CACHE	DIH
Microservices Creation	Requires coding	Low code creation and deployment of data microservices
Data Validation	Not included. Requires coding at the application level	Out-of-the-box data validation and cleansing features

### 1.3. What's the Difference Between an Operational Data Store and a DIH?

The Operational Data Store (ODS) concept is not new. It's been around for a couple of decades, continuously evolving with technological innovations, and is perceived differently by different people. In a nutshell, an ODS aggregates transactional data from multiple sources. Traditionally the ODS was designed and optimized for operational reporting and was refreshed on a daily or sometimes an hourly basis. Usually, the ODS only stores a short-term window's worth of data.

The main benefit of an ODS is its ability to consolidate data from multiple sources. Many organizations use different SoRs to manage various aspects of their data. Reporting on each data source separately provides a siloed view of the data. The ODS allows for reporting across multiple systems of records for a more complete view of the data. In addition, some SoRs offer limited reporting capabilities, so an ODS is a way for users to gain more comprehensive reporting by opening reporting capabilities to a broader audience within the organization. The ODS also enhances database security, by restricting access to a select few users.

Traditional operational data stores are not optimized for real-time API serving, due to high latency and limited concurrency. More importantly, the ODS does not have a real-time replication of the relevant data. This is fine for reporting use cases, but not

sufficient for digital applications. In contrast, the DIH is designed to support continuous innovation via an architecture that enables agility, where microservices are decoupled from SoRs, as well as from each other.

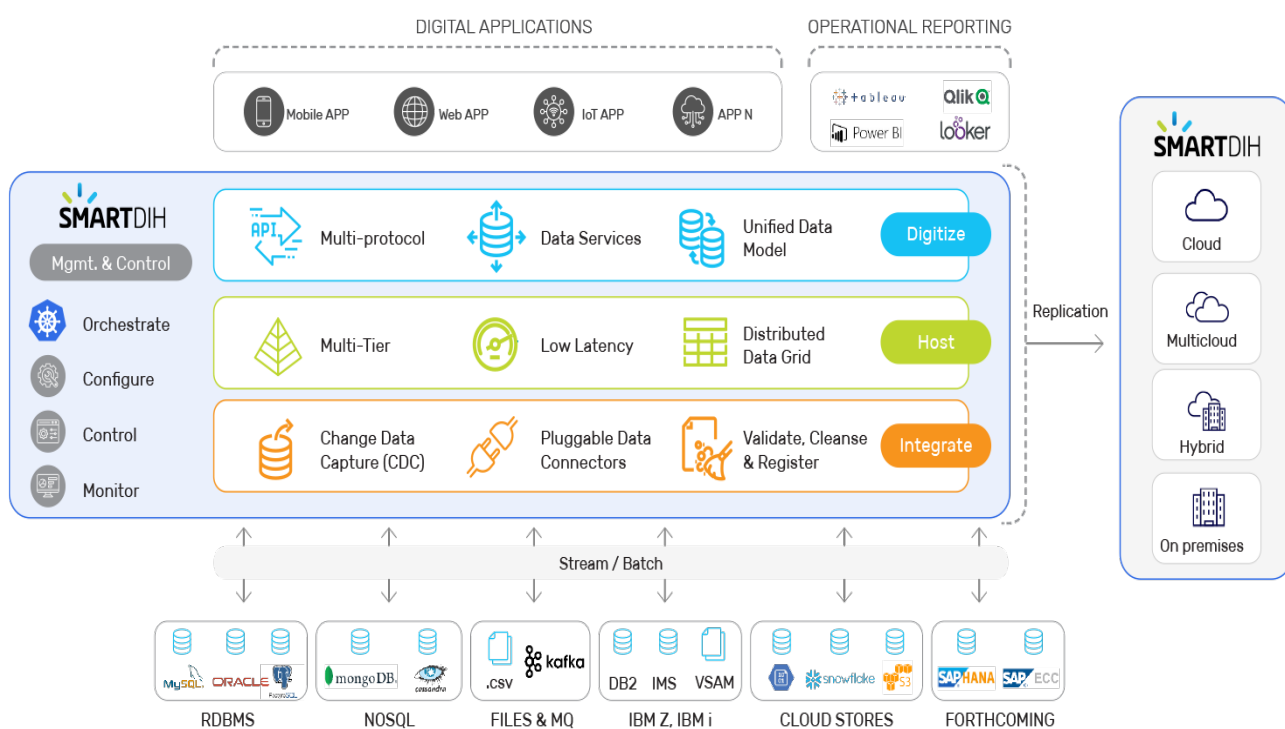
It is important to note that many organizations that are planning to implement a DIH architecture are still referring to it as ODS for historical reasons.

### Summary of key differentiators between a traditional ODS and a DIH

	Operational Data Store (ODS)	Digital Integration Hub (DIH)
Objective	Optimized for analyzing data through reporting	Optimized for real-time APIs serving to power digital applications
Data Accessibility	Usually via interfaces such as JDBC	Usually via data microservice APIs over multiple protocols
Data Integration	ETL batch integration	A combination of real-time event-based integration (CDC) and batch ingestion (ETL)
ACID Compliance	Not ACID-compliant since it's not necessary for reporting	ACID-compliant to support critical digital applications
Data Freshness	Once a day, or sometimes hourly	Real-time replication of systems of record data
Availability	Not designed for always-on availability	99.999% availability to support always-on digital applications

## 1.4. What Is Smart DIH?

Smart DIH is GigaSpaces' flagship data platform. It's designed from the ground up as an out-of-the-box Digital Integration Hub to accelerate digital transformation and power business innovation. Smart DIH enables enterprises with multiple systems of record and massive amounts of data in disparate systems - including legacy databases and cloud-based data stores - to optimally leverage their operational data for digital applications across any environment. It enables decoupling digital applications from the SoRs as well as from each other, thereby increasing agility and ensuring always-on services.



Smart DIH architecture

Smart DIH consists of four main layers, pre-integrated and available out-of-the-box:

**Data integration layer:** Offers event-based data ingestion from source data stores. Creates and manages data pipelines, applies data cleansing and validation policies, and places data in a Space.

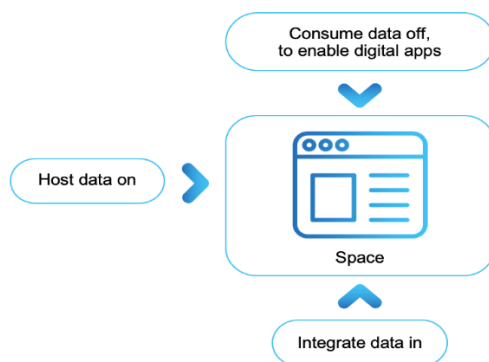
**Hosting layer:** Provides partition management, high availability, and scales to meet changing demand volumes. Storage tiering management automatically places data in the appropriate storage tier (e.g., RAM vs. SSD) and offers data indexing.

**Digitization layer:** Offers access data via APIs, microservice lifecycle management, and data enrichment.

**Management & Control:** Provides deployment, orchestration, monitoring, configuration, and control.

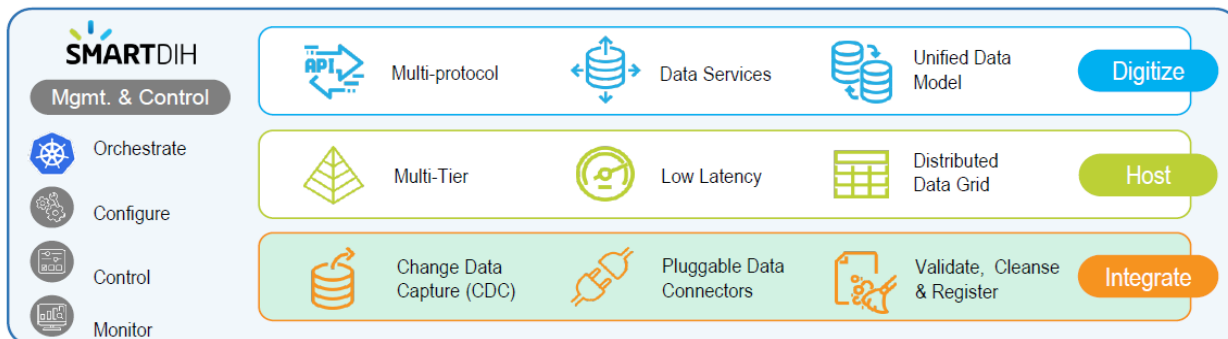
## 1.5. What is a Space?

A Space is at the heart of Smart DIH. Similar to how a database schema is a collection of tables, a Space is a managed collection of **object types**, whereas an object type is a well-defined and managed collection of **objects**, and so could be logically looked at as a database table. An object could be considered as the logical equivalent of a record.



*A Space*

## 2. Data Integration Layer



### 2.1. Objectives

The main objective of Smart DIH's data integration layer is to shorten time-to-value and reduce operational costs and efforts, by standardizing the integration from different sources. The primary user of the data integration layer is the data engineer.

### 2.2. Data Pipelines

Data pipelines are at the heart of data integration. A data pipeline is an entity responsible for importing data from data sources onto the Space, in an event-based manner. Events may either be based on changes to the data (new or updated data) or on time (schedule) and are defined based on the nature of the data (quickly or slowly changing). Fast-changing data will likely be streamed in, while slowly changing data will likely be updated either in full (e.g., small tables such as exchange rates or risky countries), or in incrementally, with only new or updated data being flown through the pipeline.

A single pipeline may be responsible for a bundle of tables. Tables are bundled together either due to technical inter-dependency or due to business-level links. The information in the pipeline configuration includes the connectivity to data sources (through a connection provider), data extraction logic (tables, filters), validation and

cleansing logic, and a representation in the DIH Space (e.g. data type mapping). The pipelines framework provides out-of-the-box capabilities for no-code definition, control and monitoring.

### 2.3. Functionality

- **Define and control data pipelines** including initial definition, view configuration, edit configuration, control, and monitoring.
- **Apply configuration parameters** including connector definition, tables, columns, and rows to be ingested, rename field names to create a unified data model, define validation and cleansing policies, define routing key, object type key (ID) and indexes for target tables.
- **Control data pipelines** including initiating a full sync, starting online pipelines, and pausing/resuming pipelines for maintenance.
- **Monitor data pipelines** including pipeline status and pipeline operational statistics and trends, such as data volume in each time unit.
- **Apply change management flows** to data pipelines, e.g., to add a new table or column.

### 2.4. IBM InfoSphere Data Replication Integration

Smart DIH's out-of-the-box integration with IBM InfoSphere Data Replication (IIDR), provides seamless integrations from data sources such as DB2, IBM z/OS mainframe, DB2 on IBM iSeries (AS/400), Oracle, and MS SQL Server. It can be utilized simultaneously with other third-party connectors that an organization uses.

GigaSpaces data integration capabilities ensure business continuity even during updates, or when common loading errors occur, with no downtime of digital services.

## 2.5. Pluggable Connectors

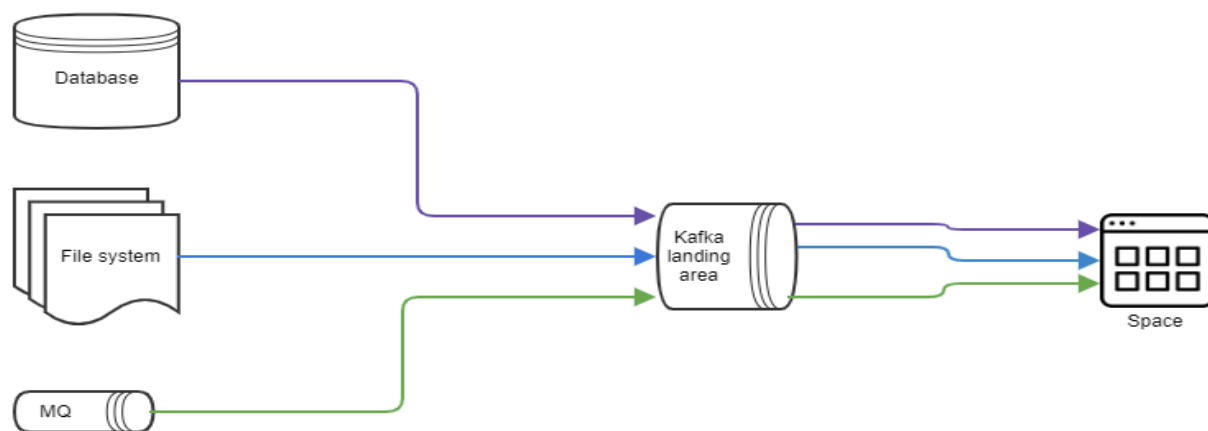
The pluggable connectors framework is the foundation for defining and operating data pipelines. Currently it is offered for streaming-based connections and will be expanded to also support batch-driven connections in future releases. It supports both strongly and loosely coupled integrations.

This framework has several benefits:

- Connecting with home-grown or existing third-party connectors
- Standardization of data pipelines, leading to lower operational costs
- Supports easy expansion of Smart DIH out-of-the-box connectors

## 2.6. CDC Based Pipelines (Change Data Capture)

CDC based pipelines consist of three parts: connector, landing area (Kafka topics into which the data is written by the connector), and ingestion flow. Data pipelines are one-way replication flows, and do not address write-through or write-behind to the system of record.



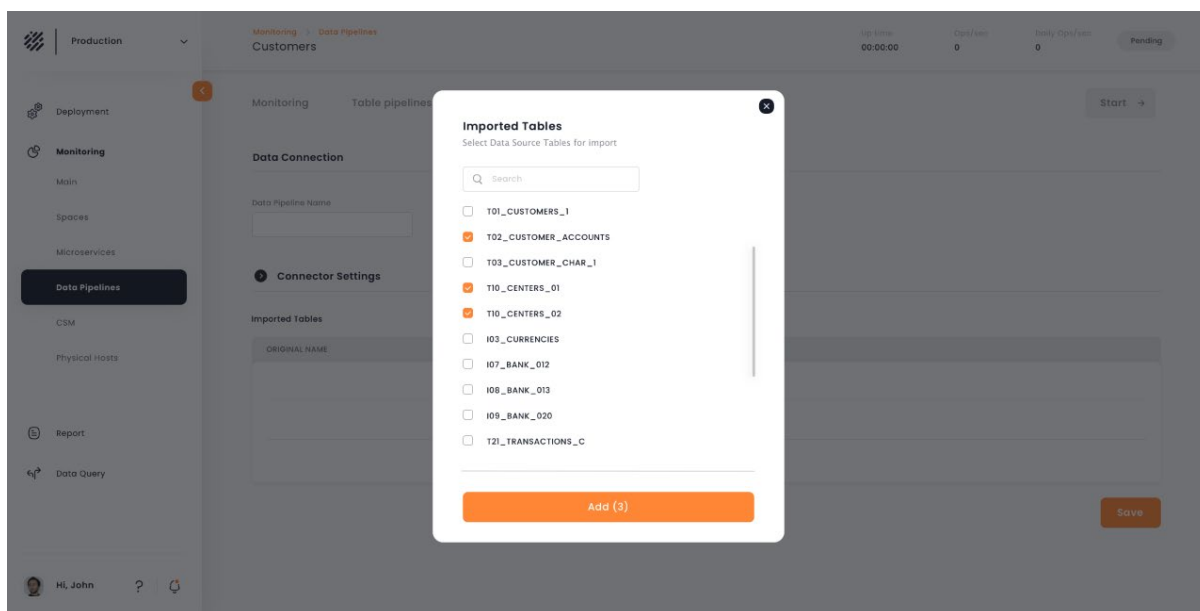
*Smart DIH CDC based data flow*



## 2.7. Addressing Changes in Data Structure

Occasionally, the data structure in the systems of record changes. The change may be within an existing table, such as a new or changed column, or it may mean that new tables have been added. In addition, sometimes a previously omitted table or column may be added.

One way to address a new table is to define a new pipeline. However, when there is a relationship between tables that requires that they are to be synchronized together, the tables need to reside in the same pipeline. Thus an existing pipeline needs to be updated to add the new tables. Business continuity is achieved during these operations and during a data schema change, such as adding a new table to a data source, as no downtime occurs.



*Adding Tables*

## 2.8. Data Validation and Cleansing

While relational databases are quite strict in enforcing the adherence of data to a predefined schema, using common schema-less object stores such as MongoDB is a

way to work around this. This however does not resolve the data validation issue, but rather just pushes the responsibility from DBAs to developers who need to address this at the application level.

Smart DIH offers out-of-the-box validation and cleansing capabilities to ensure data quality while allowing developers to focus on business logic rather than on data quality. These capabilities are agnostic to the type of pipeline and apply equally to streaming data as to batch data. The following data validation scenarios are forthcoming, or will be supported in future Smart DIH releases:

Scenario	Possible Reasons	Handling Policies	Availability
Extra columns	New column added at source, or user error	Ignore column, reject record, stop pipeline	Forthcoming
Missing columns	Semi-structured data source (e.g., JSON), or user error	Pad with empty value, pad with user configured default value, reject record, stop pipeline	Future release
Type mismatch	Mismatch between source and target data type mapping (user error)	Replace with empty value, replace with user configured default value, reject record	Future release
Value out of range	User error	Clear invalid value, replace invalid value with default value, ignore	Future release

## 2.9. Data Registration

Data quality, data freshness and data governance are essential to understand and measure the impact of the data on business processes. To provide transparency into the data journey, metadata will be generated at each step of the way.

We call the process of generating that data *Registration*.

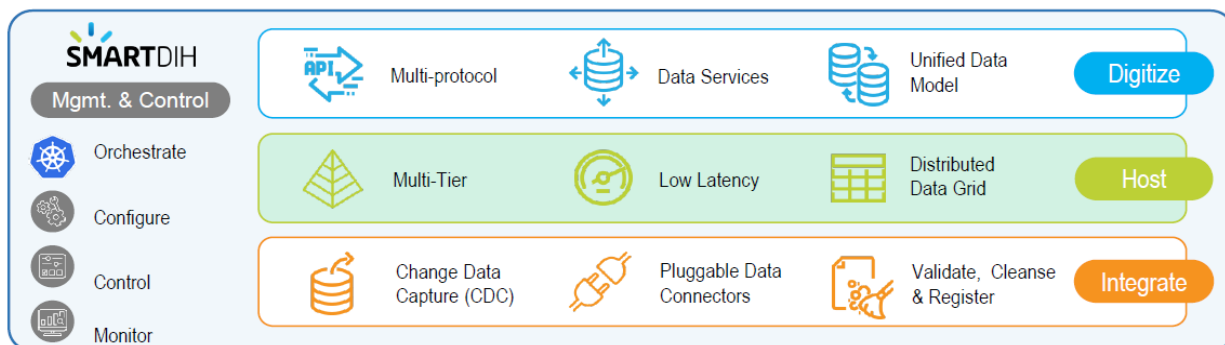
Data registration can enable:

- **Observing** – monitoring, reporting, and troubleshooting the impact the data has on achieving business and operational goals.
- **Segmentation and permissions:** restricting access to data based on the metadata describing it (e.g., country of origin).

\* Data registration will be introduced in forthcoming Smart DIH releases.

For more in-depth information about Data Integration, refer to the 'GigaSpaces Data Integration Technical Paper'.

## 3. Hosting Layer



### 3.1. Objectives

The main objective of Smart DIH's hosting layer is to provide a high performance, highly available and resilient data store for the use of near real-time digital services. It's designed to balance cost and performance, enabling business growth and handling varying levels of data consumption. It provides a unified and easy to understand data model for the use of digital application developers, supporting a mix of structured and semi-structured data, and data enrichment.

The primary users of the hosting layer are data engineers, responsible for bringing the data into the data space, and service developers, who write digital services that consume that data.

### 3.2. Functionality

For data engineers, Smart DIH provides the following features and benefits:

- Manage the unified data model at a holistic level
- Define data events to be consumed by digital services
- Define multiple indexes
- Define data tiering criteria
- Analyze data consumption trends to enable proactive scaling

For service developers, the following features and benefits are provided:

- Review of a unified data model
- Explore data
- Run direct queries to develop the service logic
- Query data through developed service
- Assess performance to uncover improvement opportunities

### **3.3. Data Modeling**

Smart DIH data is stored within a Space.

[A DIH Space](#) is a data store that hosts data from multiple sources, consolidated as a unified data model. It contains representations of replicated source tables that may be cleansed, filtered, or enriched. In addition, the data model can contain new data types fed from replicated source tables.

### **3.4. Viewing a data model**

When viewing a data model, a user can see the different object types, the object type's schema, reference to the replicated source tables, routing keys, data tiering logic, and data events.

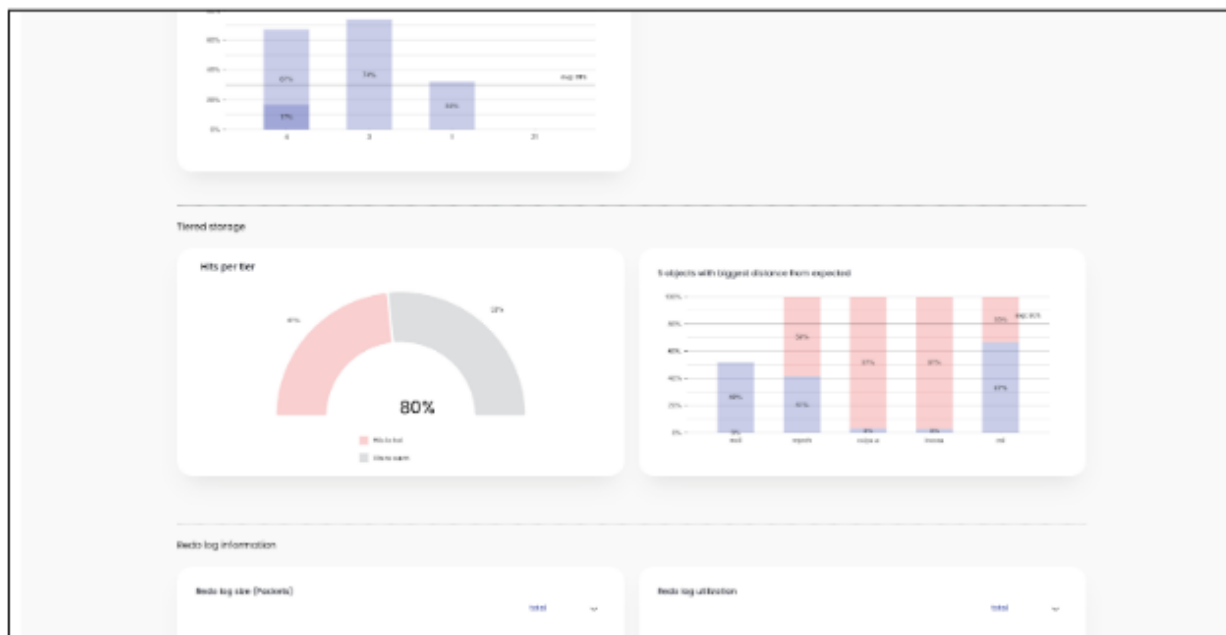
### **3.5. Editing a data model**

When editing a data model, a user can edit object type name and description, schema definition, data tiering logic for object types, and routing key.

### **3.6. Data Tiering**

Data Tiering allows enterprises to balance cost and performance by storing data on an attached local storage such as SSD, per partition. Since storage is cheaper than RAM, combining them allows placing the high priority data in RAM for super-fast response

time, while the bulk of the data that is needed less frequently by digital applications is fetched from storage. The decision about which data will be placed in RAM (i.e. the Hot tier) in addition to be placed in storage (i.e. the Warm tier) is based on business logic, in the form of simple rules. Data querying latency is a function of the coldest tier in which data resides. In other words, if even part of the data for a query resides in the Warm tier, the overall query will be running over the Warm tier.



*A DIH Space with Tiered Storage*

Local transactions are also supported in Tiered Storage, assuring consistency between different tiers in a partition and data integrity. Tiered Storage criteria also support multiple business rules. For more in-depth information about Storage Tiering, refer to 'GigaSpaces Tiered Storage Technical Paper.'

**Note:** To guarantee high performance, the user-defined tiered storage business policy needs to make sure that the majority of queries will be against the hot tier.

### **3.7. Data Exploration**

A user can explore the data using SQL queries to understand its nature, for the purpose of serving it to applications. This provides insights into:

- The expected value range for specific fields
- Whether the value set is finite or infinite in nature

### **3.8. Data Validity and Freshness Analysis**

Data validity is crucial to ensure an organization is making decisions based on accurate data, and that correct data is being served to users and customers. In addition, when data is replicated between a system of record and a DIH, it is also important to monitor and guarantee the data's freshness. It is up to the organization to define what's acceptable freshness for each type of data. For example, some data requires near real-time freshness, and each SoR data change event needs to be reflected immediately in the DIH. Other data is processed in batches, for example once a night, and thus a freshness of less than a day may be considered as acceptable for this type of data.

A data engineer can analyze and troubleshoot data validation and freshness, per a Space's object type, with visibility into:

- Rejected records
- Fields emptied or replaced with a default value
- Invalid fields
- Age of latest inserted / updated record
- Latest batch run + schedule frequency
- Average frequency of new records
- Average frequency of updated records



Some of the above capabilities are upcoming, while some will be introduced in future releases.

### 3.9. Performance Accelerators

At the heart of DIH is a high-performance data store which is key to serving digital services with instantaneous access to data as expected from digital channels. Smart DIH leverages an in-memory data grid to support extreme performance and concurrency requirements. In addition, it offers several performance accelerator capabilities to address various scenarios and boost performance even further.

- **Distributed Processing with Partitioned Space** allows for the increase of overall RAM capacity of a single space by engaging RAM of multiple nodes. It also enables parallel processing of queries by distributing the workload over multiple machines.
- **Multi-indexing Support** for accelerating read and take operations. No downtime is required to add indexes via code. Note that indexes speed up read/take at the cost of increasing footprint, and selective multi-indexing allows for read/take speed increases based upon optimum memory footprint.
- **Space-based Remoting** enables running code close to the data, thus avoiding unnecessary network latency. It also allows simple client applications to run their code on the server side, utilizing the Space's high availability and distribution processing.
- **Broadcast Object Types** overcome the high-performance cost of cross-partition joins by duplicating relevant object types across all partitions. These duplicated objects are referred to as broadcast objects. Object types suitable for broadcasting are relatively small ones, holding slowly changing data. In other words, consider broadcasting for frequently referenced lookups or dictionary tables.

### 3.10. Scaling

When workload increases, adding resources can easily be accomplished to meet new business needs. Smart DIH supports two types of scaling:

- **Scaling up/down** (vertical scaling): adding RAM and SSD resources per node (partition).
- **Scaling out/in** (horizontal scaling): adding nodes and repartitioning the data to evenly spread across old and new nodes.

### 3.11. Backup and Persistence

Recovering from a system shutdown, whether voluntary or involuntary, requires the speedy recovery of data and metadata. Smart DIH offers the following tools:

#### 3.12. Mirror

An asynchronous persistency mechanism (“write-behind”), through which data is streamed to an external data store. Such a data store may either be a relational database or an object store (e.g. MongoDB). Reference implementations are available to demonstrate the use of Hibernate to support the mirror mechanism.

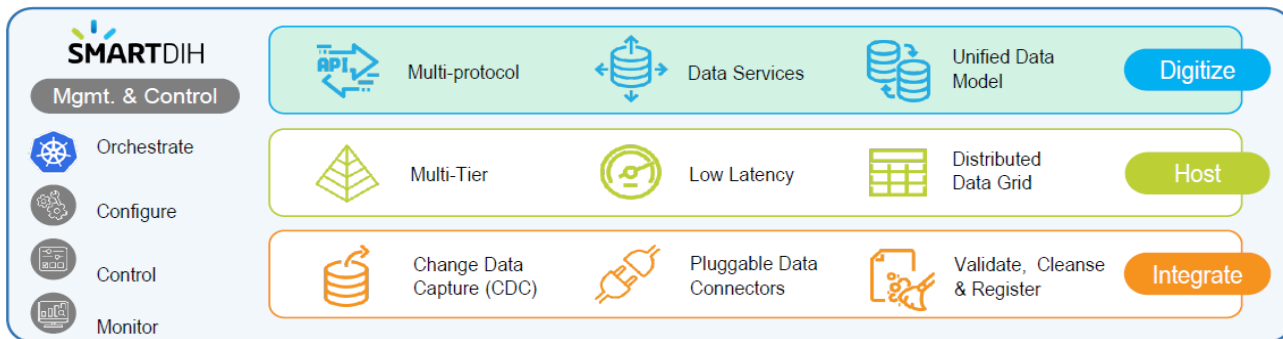
Note: The Mirror does not save time-tagged snapshots of the Space, and so it does not allow reverting to a specific point in time. Hence, the mirror is not a backup mechanism suitable for recovering from data corruption or other logical errors.

#### 3.13. Tiered Storage as a Persistence Mechanism

In addition to the benefits mentioned previously, the tiered storage configuration also provides a synchronized persistency mechanism for the Space. In this configuration, the data is first written to the Warm tier, and then the data qualified by the defined logic (rules) is cached to the Hot tier. Upon recovering from a voluntary or involuntary

system shutdown, the relevant data is automatically cached from the persistent Warm tier to the Hot tier.

## 4. Digitize



### 4.1. Objectives

The main objective of Smart DIH's digitization layer is to provide the means for continuous and fast digital innovation, whether external (e.g. customers) or internal facing. It enables operational scaling by lowering the skillset required for developing data microservices, and reduces maintenance costs by streamlining the practices used for creating data microservices.

### 4.2. Functionality

The main users of the digitization layer are the microservices developer and the business analyst.

To succeed with Smart DIH, the service developer needs to:

- Design the API through which the microservice will be invoked
- Design the query to be executed against the data
- Consider peripheral functionality the service should provide, such as relevant metrics to be recorded, to support a healthy runtime operation for the service
- Implement, deploy and test the service
- Submit the service for operationalization

In a digital ecosystem, the turnaround of a service introduction should be as short as possible, to enable continuous innovation at scale. Using the APIs and utilities provided by the digitization layer, a service developer can:

- Optimize time spent, so the majority is invested in designing and implementing according to business requirements and as little as possible on repeatable plumbing-oriented coding
- Deploy the service and expose its API for the use of digital application developers

The business analyst (acting as a product owner) is the person initiating the development of the service, to address a business need. Once the service is operationalized, the business analyst will follow-up to ensure that the expected business outcomes are indeed met.

To achieve their goals, the business analyst needs to view reports on the performance of the relevant service, for example:

- The average frequency the service has been invoked
- The number of unique users / devices by which the service was invoked
- The average record count in the service's result set
- The average latency of the service's response

### **4.3. Data APIs**

Smart DIH exposes multiple API protocols to access the data:

- Java SDK
- SQL
- .NET SDK, supporting both .NET Framework and .Net Core
- REST API

Considerations for protocol selection:

- The coverage of the different protocols. Java SDK is the most comprehensive data API protocol available for Smart DIH users. It also provides the upmost flexibility and performance. Other protocols may yet be sufficient for the specific needs, and in conjunction with other considerations may be a better choice for the specific case
- The skillset of the service developers. Increasing operational scale of new service introduction requires that the skillset of service developers does not require comprehensive Java experience. SQL is a good fit for developers whose skills reside mostly in data and less in advanced programming. A combination of SQL with Java (e.g. using JDBC) may provide a good balance between the two needs
- The organizational skillset. Organizations that are more inclined to the Microsoft ecosystem may have more .NET expertise than Java expertise
- A desire to develop in unsupported programming languages (e.g. Python, C), can be accommodated by using the REST API protocol, and have the service deployed externally to the Smart DIH cluster (yet, potentially on the same Kubernetes cluster, for network proximity)

#### **4.4. Low-Code Microservices Creation**

Microservices architecture is key for fast and continuous innovation. Central to this architecture is the independence of microservices from each other, and so microservices can be fine-grained and focused on specific functionality. As a result, a simple microservice usually exposes one endpoint, and rarely more than a few endpoints. Still, developing a simple microservice can be a complex process, for several reasons:

- Unclear business requirements

- Ill-documented data models
- Lengthy process of making the data available for developing against
- Low-quality data, requiring developers to address many edge cases
- Overhead of functionality that is not business-related. Usually, such functionality is derived from operational or regulatory-related requirements such as monitoring and auditing

Smart DIH helps reduce the complexity of developing and launching new services across several dimensions:

- Unifying the data model in the Smart DIH hosting layer provides the opportunity to properly document a single data model for the service developers to work with
- Short cycle of making the data available for developing, through the data integration module
- Applying validation and cleansing policies upstream as part of the data pipelines configuration (in the data integration layer) reduces the burden from service developers to address quality issues downstream, at the data microservice level
- A low-code blueprint for service creation is provided as part of the Digitize layer. Using this blueprint, a service developer can easily generate a Smart DIH compatible service, and package it as a deployable JAR. The overhead of coding for operational and regulatory requirements is also saved, as the blueprint automatically implements such requirements, out-of-the-box

#### **4.5. The service creation blueprint**

A blueprint is a Java project templating framework provided by Smart DIH for developers. Using blueprints is an effective way to create a repeatable project pattern, and hence shorten development cycles as well as standardize development practices



across development groups. Some blueprints (such as the Service Creation described here) are provided out-of-the-box with Smart DIH.

Using the Service Creation blueprint, a service developer can:

- Define the API for the service. The Service Creation service will automatically create a Swagger for this REST API
- Either define the service's logic as an SQL query to be run *or* choose to create a logic-free project to be implemented later using Java code

The developer *does not* need to worry about repeatable code supporting auditing and monitoring. The following code is included in the blueprint template:

- Audit logs - using a client-provided call ID, the auditor can easily trace service invocations associated with that ID. Common ways to instantiate such call IDs may be for the calling digital application client or for the user ID under whom the client is running
- Standard metrics, such as call counts, latency, count of records being returned. A service developer can easily enhance those metrics through the generated project to add service-specific ones
- Health check endpoint, through which the service can be externally monitored

## **4.6.     Microservice Operationalization and Lifecycle**

Data microservices can be deployed onto Smart DIH. Hosting data microservices within Smart DIH has several advantages:

- Executing queries close to where the data is located increases performance
- Keeping data operations on a single system enables getting a clear view of the data journey and usage, for purposes of operational reporting and observability

Once deployed, the data microservice is fully operational. A built-in web server enables direct access to the service's API, through the developer's defined port.

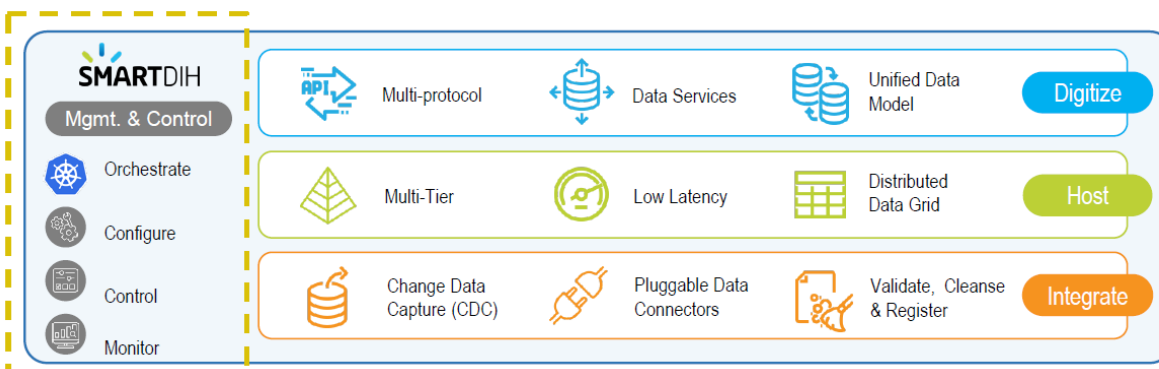
## 4.7. Event-Driven Microservices

As described in the Hosting chapter, data events can be consumed by services. The consumer service may apply one of two methods of listening:

- Notification
- Polling

Smart DIH provides a single interface for listening to events, regardless of the listening method used. This decouples between the listening mechanism, and the logic of what to do when an event occurs, as implemented by the consuming microservice.

## 5. Management & Control



### 5.1. Objectives

The Management and Control layer enables a robust integration of Smart DIH into the enterprise ecosystem. It supports business-as-usual activities such as monitoring, troubleshooting, maintenance, and change management processes. The main users are integrators, system admins, DevOps and SOC engineers.

### 5.2. Infrastructure-Level Management

Smart DIH's architecture follows the microservices pattern, and as such requires an orchestrator to facilitate the management of those microservices in an operational environment. It supports Kubernetes-based deployments out-of-the-box, using a factory-provided Kubernetes operator. The operator provides a Kubernetes cluster with the logic of how Smart DIH should be managed automatically throughout its lifecycle activities, such as deployment, resource monitoring, recovery, and others.

Multiple Smart DIH clusters may be deployed on a single Kubernetes cluster, to enable different setups for different use cases. Another scenario in which a Kubernetes cluster may be shared is to support change management practices with different environments for development, testing, pre-production and production. For this scenario, it may be that lower environments (development, testing) will be placed in

one Kubernetes cluster while higher environments (pre-production, production) are hosted on another one.

Zoned Anti-Affinity allows the primary and backup partitions to be located not only in different servers, but in distinct groups (zones) of servers, so that specific activities do not interfere with each other. Zoned Anti-Affinity also facilitates high availability.

Multi-Region Replication, which GigaSpaces implements through its WAN Gateway, is supported by the Kubernetes Load Balancer.

Deployment on VMs is also supported, with basic orchestration capabilities, provided by a legacy organic orchestrator, Service Grid.

The users of the infrastructure management capabilities are:

- DevOps and/or integrators responsible for deployment and processes
- Site Operation Center (SOC) engineers, responsible for system monitoring
- System admins, responsible for maintaining the system's health and operability

### **5.3. Application-Level Management**

Application management and control capabilities are required to enable Smart DIH functionality. Such capabilities provide:

- A no or low-code generation of system-managed entities such as Spaces, data pipelines, data microservices, etc.
- Configuration of the above entities, throughout their lifecycle
- Controlling system-managed entities, such as starting the online processing of a data pipeline or scaling out of a Space
- Monitoring system-managed entities for status, such as active or inactive; application-level errors and warnings, and metrics
- Authorization (permissions, access control) and auditing of user activities

The tooling provided by Smart DIH to support the above are comprised of:

- UI-based application: a one-stop shop for all type of management activities, providing no-code user experience
- API and CLI interfaces: automation-friendly interfaces; the CLI is also scripting-friendly making it handy for integrators and DevOps-type users
- Logs: system and audit logs, available for consumption by logging tools, as per the customer preferences such as Logstash, Splunk
- Metrics are logged into a time-series DB
- Monitoring the Grafana dashboard, which can be either consumed through a system-integrated Grafana and/or an organizational one

The main user personas that are relevant for application-level management activities are:

- Data engineers (supported by Integrators)
- Service developers
- System admins
- SOC engineers

## **5.4. Change Management**

Change management is the process through which updates are made to operational systems. In most enterprises, operational systems (also known as “Production”) are not to be touched manually. Changes made to Production systems are usually through automated processes, mostly developed by DevOps and operated either by DevOps or system admins.

The change management process usually involves a few types of environments, usually between three or four. While their names may vary between organizations, it is customary to differentiate between low and high environments, with user flexibility gradually lost when moving up the environment scale. The automated process assumes that manual changes are made and tested at low environments, and the

tested artifacts are automatically promoted up the environment hierarchy until successfully landing in Production.

To properly support change management processes, Smart DIH enables artifact portability, for DevOps processes to utilize. Thus, tested artifacts can be promoted from lower to higher environments, through CI/CD processes.

## 5.5. SpaceDeck

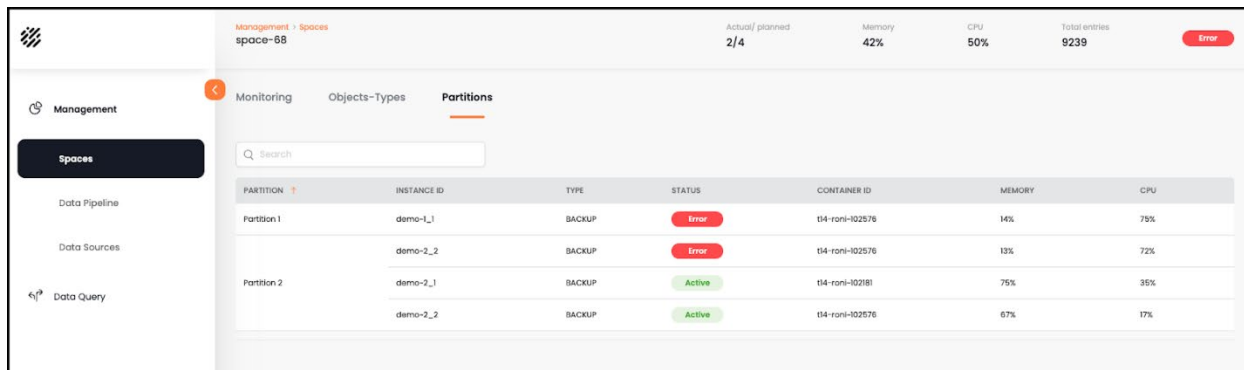
The SpaceDeck Command and Control Center for Smart DIH users provides an intuitive, streamlined user interface that currently supplements and will ultimately replace the GigaSpaces Ops Manager, GigaSpaces Management Center and Web Management Console UIs. SpaceDeck enables the definition and modification of the data which will be pulled from the SoRs and placed in the GigaSpaces in-memory data grid or in an SSD storage.

SpaceDeck offers:

- **No-code functionality:** the environment can be set up and monitored using the user-friendly SpaceDeck screens, without writing or customizing code
- **Clear flows:** for defining and running the online environment
- **Built in data and system monitoring tools:** that enable the smooth 24/7 operation of the system, such as view of transaction times and counts and spotting data bottlenecks

SpaceDeck includes:

- **Spaces:** display status, resource usage and error analysis of Spaces and related partitions

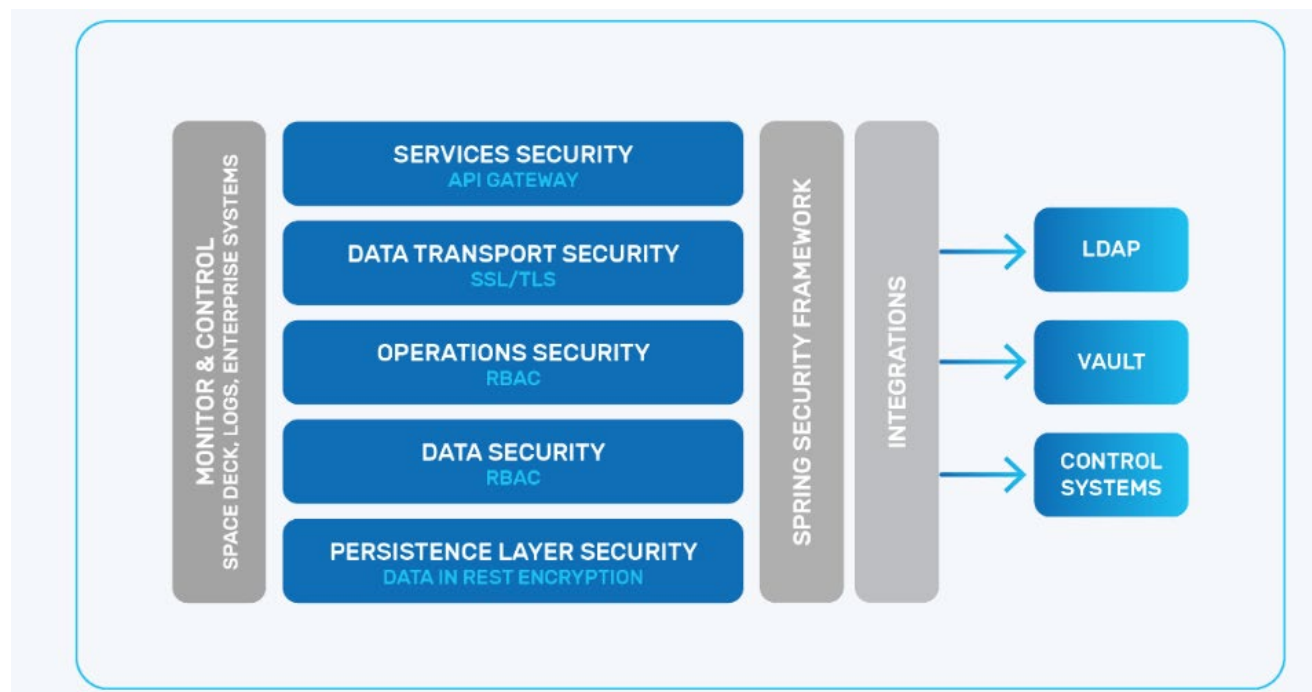


PARTITION	INSTANCE ID	TYPE	STATUS	CONTAINER ID	MEMORY	CPU
Partition 1	demo-1_1	BACKUP	Error	t14-roni-102576	14%	78%
	demo-2_2	BACKUP	Error	t14-roni-102576	13%	72%
Partition 2	demo-2_1	BACKUP	Active	t14-roni-102181	75%	35%
	demo-2_2	BACKUP	Active	t14-roni-102576	67%	17%

- **Data Sources:** specifies the data source that will be used to provide data in a data pipeline
- **Data Pipeline:** display data pipeline status details such as source table in the System of Record, resulting Object in GigaSpaces, and parameters of the data connection; allows quick one-click creation of new data pipelines
- **Data Query:** Querying the data using SQL commands, supporting full ANSI SQL-99 syntax
- **Select SQL statements can now be embedded**, enabling uniformity throughout the span of a transaction, improving data quality for business intelligence, data inquiry and ETL processes
- **Third-party tools can now automatically access** the schema configurations for GigaSpaces objects, with enhanced SQL capabilities.



## 6. Security



### 6.1. Data at Rest Security

#### On Premises

The GigaSpaces platform secures data at rest for on-prem environments utilizing Spring Security encryption functionality. The Spring Security Crypto module provides support for symmetric encryption, key generation, and password encoding. The code is distributed as part of the core module but has no dependencies on any other Spring code. Sensitive data can be kept in the Space in an encrypted format and be decrypted on the client side when the required key is supplied.

#### Cloud

The GigaSpaces platform secures data at rest for cloud deployments by utilizing native cloud encryption options such as:

- **SSE**: AMAZON Server-Side Encryption with Amazon S3-Managed Keys (SSE-S3)
- **EBS**: Amazon Elastic Block Storage Encryption

- **KMS:** Azure Key Management Service

## **6.2. Data in Transit Security (TLS)**

The GigaSpaces Secured Framework supports the native operation of TLS1.2 for Secured end-to-end HTTP Traffic using encryption and Digital Certificate authentication. The support of Kafka in the TLS1.2 protocol is utilized specifically for data integration.

## **6.3. Access Audit**

The GigaSpaces log comprises customized log files for GigaSpaces managers, GSA, and web-UI, including log levels of Warning, Info, Debug etc. These files enable auditing of client read and write operations according to defined roles and permissions.

For more information, refer to the 'GigaSpaces Cyber Security Framework' document.

## 7. Summary

Smart DIH is the world's first out-of-the-box digital integration hub that decouples digital applications from systems of record and consolidates data in a distributed in-memory data platform. With a microservices architecture and a unified API layer, Smart DIH helps accelerate innovation and the rapid building and introduction of new digital applications and services.

An integral part of enterprise digital transformations and strategies, Smart DIH offers extreme in-memory performance, autonomous elasticity and scale, business policy-driven storage, co-located microservices, low code data source integration, built-in data integration and always-on services; key, smart features which make Smart DIH suitable for an extensive range of use cases.

### About GigaSpaces

GigaSpaces is building on its in-memory computing and operational data store technologies to offer one of the market's first Digital Integration Hubs (DIH), an out-of-the-box solution that simplifies organizations' digital transformation, while drastically lowering legacy systems' TCO. Whether you need to accelerate one application with cache, or modernize your entire architecture with a Digital Integration Hub, the GigaSpaces in-memory data platform can future-proof your investment. Never before has it been this straightforward to accelerate API-powered digital applications to transform user engagement, legacy modernization, and 'Customer-360' software infrastructure projects.

Smart DIH is part of the GigaSpaces Smart suite of products, alongside the award-winning Smart Cache solution.

GigaSpaces offices are located in the US, Europe and Israel with partners such as Capgemini and Cognizant around the globe, serving customers such as Morgan Stanley, Bank of America, CSX, Goldman Sachs, Societe Generale, Credit Agricole, American Airlines, Avanza Bank, Avaya, CLSA, and UBS. For more information visit [www.gigaspace.com](http://www.gigaspace.com).