Gartner

Innovation Insight: Turbocharge Your API Platform With a Digital Integration Hub

Refreshed 28 January 2022, Published 10 July 2020 - ID G00722630 - 27 min read

FOUNDATIONAL This research is reviewed periodically for accuracy.

By Analyst(s): Massimo Pezzini, Eric Thoo

Initiatives: Software Engineering Technologies

Application leaders struggle to deliver high-throughput, responsive and reliable APIs while minimizing the workload hitting their systems of record. By extending their API platform with a digital integration hub, they can address these issues and add value by enabling analytics and data integration.

Additional Perspectives

- Summary Translation: Innovation Insight: Turbocharge Your API Platform With a Digital Integration Hub
(14 August 2020)

Overview

Key Findings

- APIs are essential to empower digitalization, but exposing an API layer above multiple systems of record can drive excessive and costly workload growth while creating a brittle body of integration code that's complex to implement and expensive to maintain.

- Accommodating such potentially massive workload expansion for some systems of record can be hard and expensive, at times even impossible.

- Creating an API services layer that enables gradual modernization of the underlying systems of record requires a more sophisticated, loosely coupled approach than the conventional architecture with which most API developers are familiar.

Recommendations

As an application leader responsible for API-based integration and architecture to support large-scale, multiexperience delivery, you should:

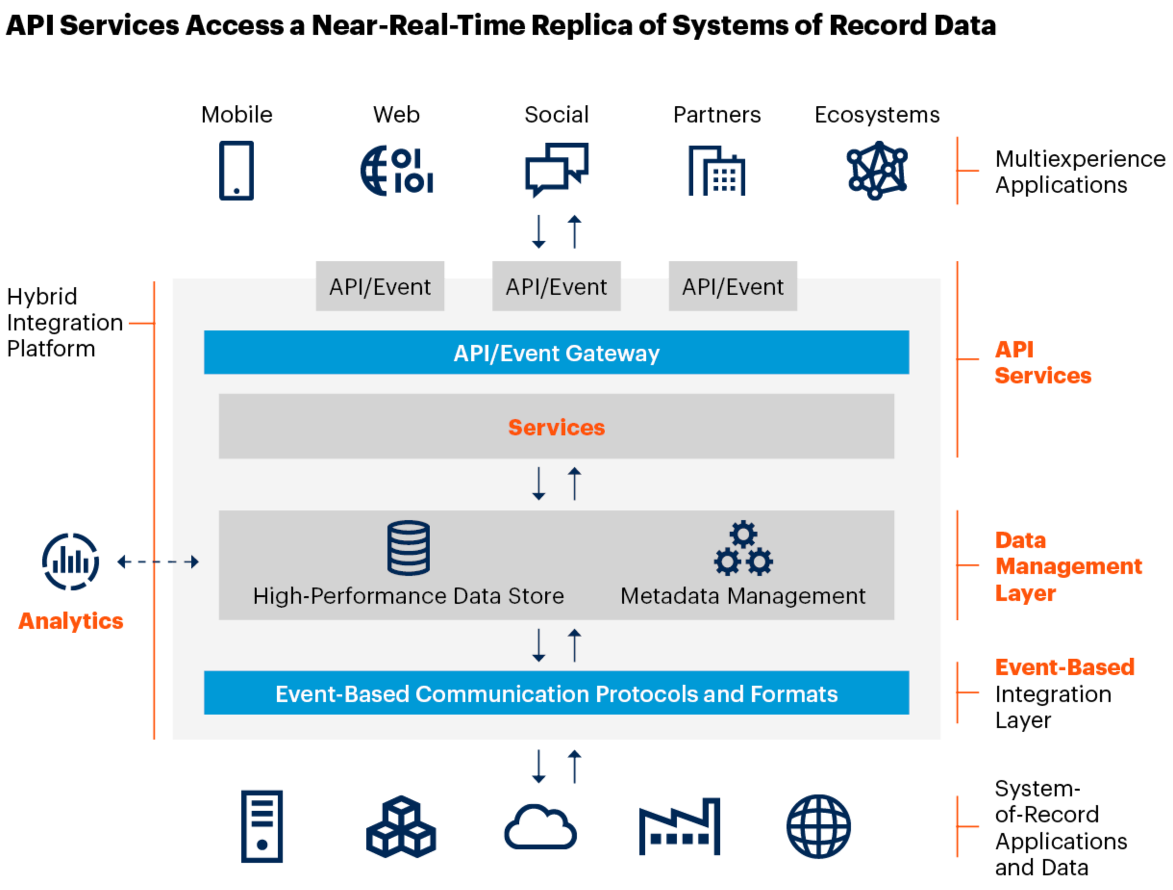
- Dramatically increase throughput and scalability while reducing latency of the API services layer by including a digital integration hub (DIH), which replicates appropriate system-of-record data. In this way, you offload expensive API-generated workloads from the systems of record, improve API performance, enable 24/7 operations and drastically decouple the architecture.
- Assess the potentially high benefits of a DIH-enabled API platform against the usually significant associated costs and efforts, and against the architectural and technical maturity of your team. Do so because:
  - In implementing a DIH, you will have to tackle complex and often unfamiliar architectural, data management and technical issues.
  - Despite a growing number of providers focused on this opportunity, DIH industry experience is still limited, comprehensive products are few, and relevant skills are hard to find and expensive.

Analysis

Application programming interfaces (APIs) are the ubiquitous enablers for digital modernization and the undisputed springboards for the "composable enterprise" (see "The Applications of the Future Will Be Founded on Democratized, Self-Service Integration"). APIs empower multiexperience, mobile apps, social networks, chatbots, ecosystem collaboration and other scenarios by accessing data hosted in "system of record" applications via a multilayered, API-enabling technology stack — that is, an "API platform" (see "Top 3 Trends in Application Architecture That Enable Digital Business").

By adding a digital integration hub (DIH) to their conventional API platform, application leaders responsible for integration architecture can deliver high-throughput, responsive and reliable APIs while minimizing the workload hitting their systems of record and improving architectural agility (see Figure 1).

Figure 1. DIH-Enabled API Platform


API Services Access a Near-Real-Time Replica of Systems of Record Data
Mobile
Web
Social
Partners
Ecosystems
Multiexperience Applications
Hybrid Integration Platform
API/Event
API/Event
API/Event
API/Event Gateway
API Services
Services
Analytics
High-Performance Data Store
Metadata Management
Data Management Layer
Event-Based Communication Protocols and Formats
Event-Based Integration Layer
System-of-Record Applications and Data

Source: Gartner
722630

Definitions

A DIH architecture aggregates data from multiple system-of-record sources into a low-latency, high-performance data management layer, usually accessible via APIs, but also via events or batch files. The data management layer synchronizes with the back ends via event-driven integration patterns. The DIH enables high-scale, 24/7 access to data while offloading systems of records and improving business agility. Organizations can obtain additional value by also leveraging a DIH in analytics and data integration use cases.

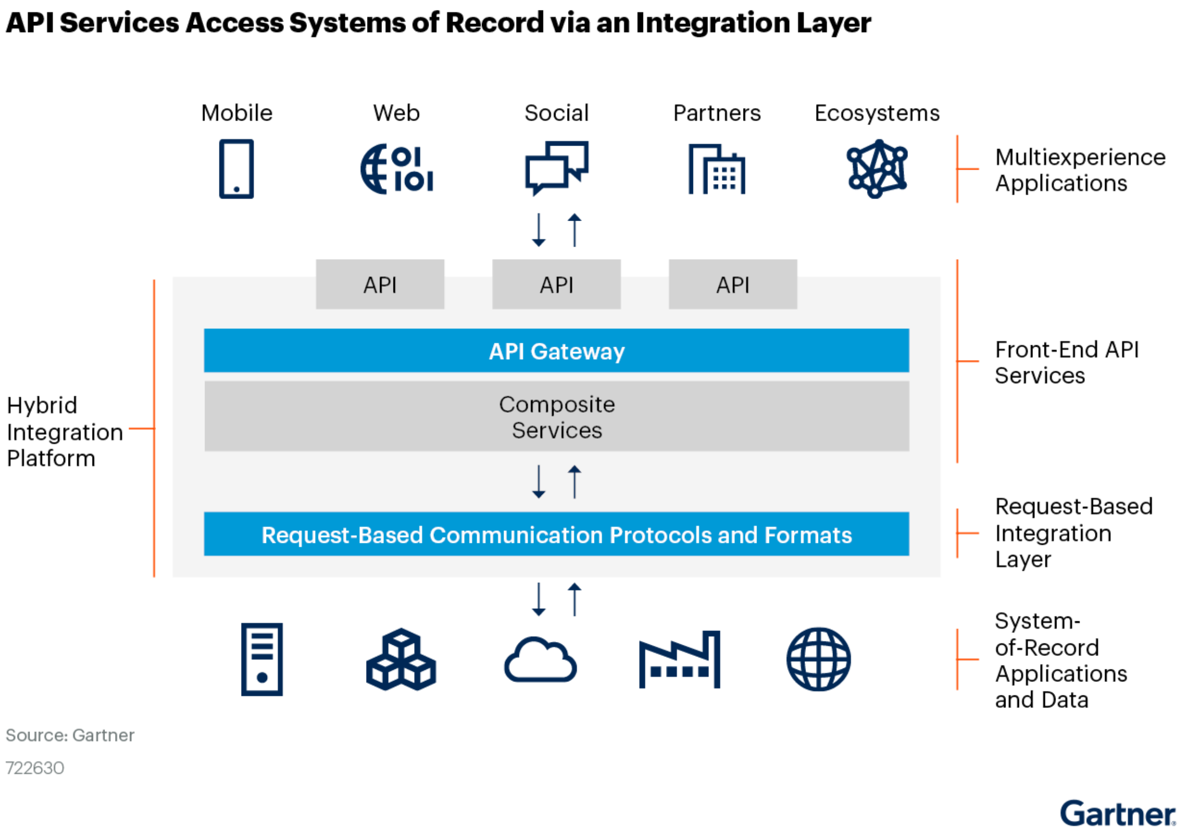
Description

Access to a dispersed set of system-of-record data via an API services layer is conventionally implemented by developing integration "composite services" — that is, chunks of "integration logic" that aggregate back-end data to implement the API's functionality.

This approach, which Gartner refers to as mesh app and service architecture (MASA), has been well-known since the early days of service-oriented architecture in the 1990s (see "Adopt a Mesh App and Service Architecture to Power Your Digital Business"). MASA is routinely implemented via an API platform that comprises an API gateway and a typically request-based mediation (or integration) layer that connects the API services with the systems of record (see Note 1 and Figure 2).

Keep in mind that Figure 1 shows (in orange) the logical components that differentiate a DIH-enabled API platform architecture from the setup in Figure 2.

Figure 2. Conventional API Platform


API Services Access Systems of Record via an Integration Layer
Mobile
Web
Social
Partners
Ecosystems
Multiexperience Applications
API
API
API
API Gateway
Composite Services
Request-Based Communication Protocols and Formats
Hybrid Integration Platform
Front-End API Services
Request-Based Integration Layer
System-of-Record Applications and Data
Gartner

Source: Gartner
722630

However, in a demanding, high-performance and high-scale digital business environment, where APIs must serve up to hundreds of millions of calls per day, the MASA approach may prove inadequate for reasons of cost, complexity, availability and lack of agility (see Note 2).

Via the data management layer, which stores aggregated (and optimized for API serving) replicas of the system-of-record data, a DIH-enabled API platform can help you address these challenges (see Note 3).

The key architectural building blocks of a DIH-enabled API platform (see Figure 1 and Note 4) are:

- The data management layer
- The front-end API services layer
- The integration layer
- An optional analytics capability
- Management, monitoring, administration and security capabilities

The DIH is not a universal repository of all data. It stores domain-specific data needed to support certain API services. Therefore, it may be necessary to implement multiple, specific DIHs to support different needs.

A DIH-enabled API platform can be seen as a way to “put data in the integration layer,” which must be supported by your HIP strategy. It is also important to note that a DIH is neither a data caching layer nor an operational data store, although it has similarities with these architectures (see Note 5).

Benefits and Uses

The use cases for a DIH-enabled API platform architecture include, but are not limited to, enabling:

- **Responsive user experience,** which accesses a consolidated, yet real-time, view of data scattered across multiple systems of record (see Case Study 1).

- **Offloading** systems of record from expensive workloads generated by the API services, thus downsizing (or at least not expanding) costly (for example, mainframe-based) systems of record or reducing access fees by limiting API calls to SaaS applications.
- **24/7, always-on API services,** even in situations where the back ends must be put offline for maintenance or upgrade. In this case, the DIH continues to provision data, although certain APIs (for example, those triggering updates to data in the back ends) may be temporarily inhibited or limited in functionality (see Case Study 2).
- **Decoupling the API services layer from the systems of record,** which is a desirable outcome for two primary reasons:
    - **Supporting legacy modernization by** making it possible to replace some system-of-record applications with minimal impact on the API services layers.
    - **Normalizing the APIs for a certain application domain,** so that with a single set of APIs the multiexperience applications can access data held in multiple systems of record (for example, ERP applications), possibly from different vendors.
- **Providing real-time business insight,** by analyzing the DIH data to offer additional services or to detect business events (for example, a supply chain disruption or a potential fraud) that require timely actions (see Case Study 3).
- **Supporting data integration requirements** by sending the DIH data to other endpoints, by applying appropriate transformation rules if needed. For example, to simplify integration between different systems of record (see Case Study 4).

The benefit of DIH broadens when applied as part of a wider scale of data hub strategy, which is an architecture aimed at providing common and coherent capabilities of practices and tooling across integration, data and analytics, alongside governance (see "Data Hubs: Understanding the Types, Characteristics and Use Cases"). DIHs add a set of complementary or embedded capabilities to existing data hubs.

The real-time, event-driven data delivery characteristic of DIH complements the emerging notion of data fabric. This optimally combines data integration and data management technologies, data and analytics governance and related architecture design and services delivered through orchestrated platforms and practices (see "Data Fabrics Add Augmented Intelligence to Modernize Your Data Integration").

Adoption Rate

The implementation of DIH-enabled API platforms is still primarily limited to large or global organizations with complex and large-scale requirements. They have the skills, as well as the technical and financial resources, needed to implement such a complex architecture.

Industries that have adopted DIH-enabled API platforms are typically those serving large consumer-type constituencies, where providing low-latency, real-time, highly available and accurate access to consolidated data is critical for business success. These industries include retail, transportation, travel, banking, insurance, telecom, utilities, consumer packaged goods, gaming and higher education.

The sophistication of these implementations varies greatly, from advanced caching layers, all the way to full implementations of the DIH-enabled API platforms, including bidirectional synchronization and analytics functionality.

However, as organizations move toward the composable enterprise, DIH-enabled API platforms will emerge as key architectural foundations, and deployments will become more comprehensive.

Factors that will facilitate and drive future DIH adoption (also by mainstream organizations) include:

- Vendors providing out-of-the-box DIH-enabled API platforms, either in the form of software products or cloud services (such as the data hub form of iPaaS).
- Vendors coming to market with applications or cloud services based on DIH-enabled API platforms, typically to support real-time, 360-degree customers, employees and products.
- IT cost-saving programs via system-of-record offloading and API access fees reduction.
- Organizations' desire to deliver responsive and real-time multiexperience to customers.
- Application modernization initiatives requiring a radical decoupling between systems to favor their gradual replacement.
- Business leaders who desire deeper and more real-time insights on their multiexperience-led businesses by leveraging event-based approaches and analytics.

A growing number of organizations will invest in DIH-enabled API platforms to support their digital business initiatives and, later, to lead them toward the composable enterprise.

Risks

In terms of business risk, the impact of a failed DIH-enabled API platform initiative can manifest in:

- An inefficient or incomplete API services layer that forces direct access to the system-of-record endpoints, thus creating duplication of efforts, more complexity and higher costs.
- Disappointing experiences for customers, employees, suppliers and citizens, which could undermine the credibility of the entire digital transformation initiative.

Unfortunately, a DIH-enabled API platform can fail because you will have to tackle several challenges. These include:

- Assembling multiple technology components, possibly from different providers.
- Monitoring, managing, administering and securing a complex, highly distributed architecture made of multiple, loosely coupled moving parts.
- Deploying “scale out” and low-latency data management technology (such as NoSQL DBMSs, in-memory data grids and in-memory DBMSs).
- Supporting bidirectional synchronization between the high-performance data store and the system-of-record endpoints by combining event-driven approaches with Command Query Responsibility Segregation (CQRS) patterns.
- Developing API services that can tolerate an event-based “eventual consistency” between the data in the DIH and in the systems of record.
- Rebuilding in the API services the data integrity rules implemented in the systems of record.
- Designing the domain-specific data models for the business entities in the DIH.

The compounding of these challenges exposes you to risks of failure or suboptimal outcomes if addressed without proper architectural and technical analysis and skills building. Nor should you expect much support from service providers, given the industry's still limited experience with this architecture.

Sharing and reconciling metadata across these different tools to facilitate common operations is another key challenge. Failure to consider this factor can expose a DIH initiative to multiple risks, including:

- Lack of openness in the supporting tools, which inhibits business analysts' ability to discover data across multiple integration platforms or event streams.
- The need to understand the structure of (or to interact directly with) the underlying applications and data environments due to the limited metadata-driven discovery and examination of abstracted interfaces to data sources and application flows.

This issue is particularly sensitive if APIs are not sufficiently abstracted from the underlying applications and data environments, so that complex underlying data models are surfaced in API definitions. To avoid this problem — often referred to as "data model dumping" — you should use an "API first" approach based on the understanding of the consumer application requirements. This would ensure the data design supporting APIs is not for one-off, narrow use, but instead harnesses metadata to expose relevant data for multiple consumption scenarios.

- An inability to monitor, track, analyze and share metadata, predicated on having an inventory of data assets and catalogs. Often, this is a challenge when organizations are overwhelmed by silos of project-specific approaches in which data flow metadata is not visible across projects. To avoid this issue, you should ensure semantic consistency of diverse data and application flows via shared metadata. Shared metadata prepares the DIH-enabled API platform for repurposing of data for future usage, such as real-time analytics, and enables data integration/preparation (see "Augmented Data Catalogs: Now an Enterprise Must-Have for Data and Analytics Leaders").

Recommendations

As an application leader responsible for API-based integration and architecture to support large-scale API delivery, you should:

- Dramatically increase throughput and scalability while reducing latency of the API services layer by including a DIH, which replicates appropriate systems of record data. In this way, you offload expensive API-generated workloads from the systems of record, improve API performance, enable 24/7 operations and drastically decouple the architecture.
- Carefully assess the potentially high benefits of a DIH-enabled API platform against the usually significant associated costs and efforts, and against the architectural and technical maturity of your team. This is because:
    - By implementing a DIH you will have to tackle complex and often unfamiliar architectural, data management and technical issues.
    - Despite a growing number of providers focused on this opportunity, DIH industry experience is still limited, comprehensive products are few, and relevant skills are hard to find and expensive.

For a more detailed set of recommendations about when to adopt a DIH-enabled API platform, please see Note 6.

Representative Providers

Providers with an out-of-the-box DIH-enabled API platforms:

- GigaSpaces (InsightEdge Platform)
- Hazelcast (Hazelcast Jet)
- Mia-Platform (Mia-Platform)
- SAP (SAP Cloud Platform Digital Integration Hub for Industry Transformation)

The following providers have products or cloud services that include some or most of the DIH building blocks.

Providers combining data management, event brokering and stream analytics:

- Fincons Group (Fincons Fast Data Lake)
- GridGain (GridGain In-Memory Computing Platform)
- Pivotal Software (GemFire + Greenplum Connector)

- Radicalbit (RNA)
- Striim (Striim Platform)

Providers combining integration and data management capabilities:

- Bouvet (Sesam)
- Cinchy
- Dell Boomi (Boomi AtomSphere, Master Data Hub and Flow)
- ForePaaS
- Informatica (Informatica Data Integration Hub)
- Maestrano (Maestrano Hub!, Impac!, Connect!, Reconcile!, Nex! and Presen!)
- OpenText (Alloy)

Providers with in-memory-enabled data virtualization technologies:

- Denodo (Denodo Platform)
- eQ Technologic (eQube Platform)
- Stone Bond Technologies (Stone Bond Enterprise Enabler)

Providers with DIH-based software or cloud services:

- Segment
- Tealium (Tealium AudienceStream CDP)

Providers offering most of the DIH components as part of their “megaPaaS”:

- Amazon Web Services
- IBM
- Microsoft
- Oracle

- Salesforce (including MuleSoft)
- Software AG
- TIBCO Software

Case Study 1: Example of DIH-Enabled API Platforms Improving API Delivery (Centrica)

Centrica is an international energy services and solutions provider that serves several millions of accounts mainly in the U.K., Ireland and North America. It is the parent company of British Gas.

Over the past several years, the British Gas unit's SAP application team developed approximately 100 to 120 SOAP and OData APIs to enable access to customer, product, contract and other data handled by the company ERP (SAP IS-U) and CRM (SAP CRM) system-of-record applications. These APIs were being used by a range of web and mobile applications, developed by the company's digital team. Millions of private and business British Gas customers used these applications, which generated between 5 and 7 million API calls per day into the systems of record. But these APIs were often redundant and were implemented on a complex architecture combining SAP Process Integration and SAP Gateway. This arrangement led to suboptimal performance, with some API calls requiring several seconds to respond and in some cases timing out after 30 seconds. The arrangement also meant the system lacked 24/7 availability as the systems of record had to be periodically shut down for maintenance or other reasons.

In early 2019, the Centrica global integration team took over the API implementation responsibility and developed a new cloud-based, DIH-enabled API platform. The platform includes SAP HANA as the high-performance data store (whereas SAP SLT is used for near-real time data replication from SAP IS-U and SAP CRM). The platform also incorporates the SAP CAP framework, which developers use to construct the new APIs, according to an API-first, OpenAPI-based model.

Even if it took quite some time to set up, the new architecture is providing remarkable benefits. It delivers 24/7 availability, as data is available even when the systems of record are down, and performance has improved dramatically. In a benchmark study, Centrica showed how an API that 95% of the time would provide a response time of 1.8 seconds, once redesigned for the new platform now takes 60 milliseconds with a throughput of 100 API calls per second. Moreover, the team estimates this new approach will allow it to reduce by about 75% the amount of APIs needed to enable the digital applications. Finally, because of the new architecture, the digital applications are now almost completely decoupled from the SAP ERP and CRM applications, which will make the system-of-record replacement easier.

An initial set of APIs, which supports more than 50% of the total workload, is already in production, and the team expects to complete all the planned API development by YE20. However, the legacy APIs will remain in operation until well into 2021, while the digital applications gradually migrate to the new environment. Based on the British Gas success, Centrica is now planning to deploy the DIH architecture in other subsidiaries.

Case Study 2: Example of DIH-Enabled API Platform Supporting 24/7 Operations (Helvetia Assicurazioni)

Helvetia Compagnia Svizzera d’Assicurazioni is the Italian branch of the Helvetia Group, an all-lines insurer headquartered in St. Gallen, Switzerland, and also operating in Germany, Austria, France, Italy and Spain. The company has strengthened its position in the market through important corporate acquisitions. Helvetia Italia Group maintains its standing among the best players in the insurance market. Helvetia’s insurance solutions cover a spectrum of products for private individuals and businesses, as well as bespoke insurance solutions and reinsurance in selected niches worldwide.

In 2017, the company started a strategic digital transformation initiative aimed at: (1) improving agility and time to market for new products; (2) offering a simplified and responsive omnichannel experience to its 4 million Italian clients; and (3) enabling new channels and legacy modernization. At the same time, the company was completing the acquisition of two smaller insurance firms; thus, it needed an architecture that could support the consolidation of the relevant system-of-record applications.

Like most insurance companies, Helvetia lacked an integrated customer view, as data about its clients was scattered across a plurality of custom and packaged applications supporting life, nonlife, reinsurance, claims and other business functions. Some of these systems of record, moreover, proved unable to provide the responsive experience mandated by mobile and internet channels because of the company's nonscalable, legacy architecture.

Therefore, in January 2019, Helvetia opted for implementing a DIH-enabled API platform based on the Mia-Platform Suite, a cloud-native environment deployed on Google Cloud Platform. The Helvetia API platform now supports bidirectional synchronization between the systems of record and the Mia-Platform data management layer (MongoDB-based) via a combination of change data capture (CDC) and message queuing technology. In this way, the architecture completely decouples the Kubernetes-based API services supporting the digital channels from the systems of record, thus delivering fast response time (less than 65 milliseconds for customer data query) and enabling 24/7 availability for digital operations. The latter is a requirement that will become more important in the near future, and may sometimes even be fundamental, to support the integration of partners' services and customer direct touchpoints management (mandatory in Italy since July 2020).

The decoupling is also helping Helvetia to migrate the acquired insurance companies' customers to the company system-of-record applications in a smooth way without compromising the user experience. Currently, Helvetia's DIH-enabled API platform integrates about 10 system-of-record endpoints and supports the company's contact center. It will be soon extended to support the customer portal (integrated into the home insurance division of the company).

In 2021, Helvetia's DIH platform will be ready to support faster integration with partners and banks, and to enable mobile applications for the insurer's agents and customers. The architecture will enable the company to improve customer services by integrating capabilities from a variety of business partners including Europ Assistance Italia, SIA (payment processing), CoffeeBean (identity and access management) and FDM Group (optical character recognition services for claims management), switching from a legacy data flow and batch integration to real-time operations.

The new architecture has been developed according to a step-by-step plan, together with the progressive introduction of agile methods and DevOps practices to some of the company's core systems. Helvetia aims to extend this approach to its strategic applications. This phased method has enabled Helvetia to define its new architecture and complete the first phase of this ambitious transformation in less than two years, providing two working business cases already. It is using the experience to strengthen the cooperation between business and IT and start shifting its legacy culture toward a modern business-IT strategy.

Moreover, the new architecture will enable the company to develop complex products and services that once required too much time, thanks to the reusability of the components, real-time aggregation and 24/7 availability of all users' data. The next phase of the DIH implementation will be focused on reducing time to market for new products and to improve flexibility of new developments.

Case Study 3: Example of DIH-Enabled API Platform Delivering Innovative Client Services (Macquarie Bank)

Macquarie's Banking and Financial Services Group — the retail banking and financial service business within Macquarie — identified the need to build a smarter digital banking platform and mobile app that could help it continually evolve and build a more personalized and intuitive experience for its Australian retail banking customers.

This was achieved by implementing a DIH-enabled API platform that aggregates customer-related data from a variety of internal systems of record. The data is then made available to mobile apps and web applications for clients, as well as to white-label partners that want to incorporate Macquarie's services into their own offerings. The bank also supports an open banking API strategy, which was the first of its kind in Australia.

The digital platform, which is deployed in a public cloud environment, is based on APIs supported by an API gateway, and the microservices-based architecture is deployed in containers. Orchestration is managed by Kubernetes. The platform uses a NoSQL DBMS (Apache Cassandra) to implement the data management layer and Apache Kafka to synchronize with the system-of-record endpoints. Through the use of an event stream platform (Apache Spark) and a search capability (Apache Solr) against the data management layer, the bank has been able to deliver innovative services to its clients. These include a natural language search in the customer transaction history, customer self-service analytics, and personalized recommendations and services.

Case Study 4: Example of DIH-Enabled API Platform Supporting Data Integration (Utility Company)

For this North American company, having up-to-date data on employees and temporary workers is extremely important. Different operating units need to quickly and reliably determine whether, where and when the skills required to accomplish operational tasks (for example, deploy a service to a customer) are available. However, the relevant information was scattered across three different systems of record: Workday (which stores employees data), SAP Fieldglass (temporary workers data) and a third custom application (consultant and supplier worker data). Moreover, different operating units wanted to receive the worker data according to different modalities (for example, via files, REST APIs or web services calls). Hence, directly integrating a significant number of downstream applications with the three systems of records was deemed a suboptimal approach.

The HCM department of the company therefore decided to implement a DIH architecture that collects data in real time from the three systems of record and stores them in an Oracle DB in an aggregated and standardized format. This database provides a consolidated and up-to-date view of all the workers, both employees and temporaries, in a single worker data hub. The team used the Workato integration platform as a service (iPaaS) to synchronize the systems of records with the hub. Similarly, the downstream applications in the operating units use Workato to download the data they need, when they need them, using whatever protocol they want. Notably, these application teams developed the integration flows moving the data from the worker data hub in a self-service fashion, with the support of the worker hub team.

Today, approximately 20 downstream applications consume data from the hub, but several more are expected to be integrated in the future.

Acronym Key and Glossary Terms

CDC	change data capture
CQRS	Command Query Responsibility Segregation
DBMS	database management system
DIH	digital integration hub
HIP	hybrid integration platform
iPaaS	integration platform as a service
ODS	operational data store
RDBMS	relational database management system

Note 1: How a Conventional API Platform Works

A conventional API platform connects the API services layer to the system-of-record endpoints via a request-based integration layer to implement “composite services” for the APIs. These services orchestrate transactions in the systems of records to implement the application logic associated with the different APIs. A HIP is often used to provide the necessary integration capabilities.

Note 2: Limitations of the Conventional API Platform Architecture

- **High cost:** A huge number of API calls can imply a massive, often low-value workload hitting the system-of-record endpoints, thus requiring significant investment in infrastructure or in SaaS API access fees. At times, supporting this workload may even imply a radical redesign of the system-of-record applications, often a simply unacceptable proposition.
- **Complexity:** If the back-end data is highly fragmented across multiple applications and sources, the relevant integration logic could be extremely hard to develop, deploy, run and manage.
- **Availability:** If some back-end systems are unavailable — e.g., for maintenance — or if an integration link is not working properly, some API services will not function, thus compromising business operations. This is an unacceptable situation in today's 24/7, always-on, digital environment.
- **Lack of agility:** A change in a system-of-record endpoint (for example, a modification of a field in a database table) may imply some reengineering or even a redesign of the composite services.

Note 3: Examples of DIH-Enabled API Platforms Supporting Business-Critical Scenarios

- A consumer electronic company in Europe implemented a proto-DIH architecture to offload its SAP ERP from the massive inquiry workload generated by its B2B e-commerce portal. Before the DIH was deployed, the portal was pushing more workloads into SAP ERP than the rest of the enterprise.
- A railway company in Asia implemented a DIH architecture because its original setup could not support the dramatic increase in its back-end system workload coming from the online reservation and ticketing applications. Before the DIH was implemented, this massive transaction volume resulted in system unavailability at critical peak times.
- A government portal in the U.S. had to be reengineered according to a DIH-like architecture because the original design could not support the unexpectedly high workload coming from citizens who wanted to access the service. Before the DIH was established, the portal was plagued by crashes and unavailability, thus risking undermining the credibility of the whole initiative.

Note 4: Anatomy of a DIH-Enabled API Platform

A DIH-enabled API platform consists of the following conceptual layers:

- **The data management layer,** which includes two components:
    - **High-performance data store:** This holds a copy of proper subsets of back-end data, providing consolidated and domain-specific “views” of business entities (e.g.,customers, products, prices, itineraries, students and patients) optimized for API servicing. It can be implemented in many ways, e.g., via a regular RDBMS or, for more demanding scenarios, via scale-out data stores, such as in-memory computing platforms (typically in-memory data grids or in-memory DBMSs) and/or NoSQL DBMSs.
    - **Metadata management:** This provides a metadata-based inventory of data assets and capabilities to discover, capture and synchronize metadata models with back-end applications. This component allows capabilities to be reused for diverse types of data flow and insulates front-end data consumers from the complexity and structure of the underlying applications and data environments.

It can derive physical/logical models, discern and reconcile relationships, track lineage and impact, share and synchronize metadata with other tools, and enable metadata-driven development and introspection (for example, to support domain-driven design approaches).

- **The front-end API services layer:** This implements the data access functionality, typically via REST APIs, but also via other mechanisms (e.g., OData, GraphQL, event channels or batch interfaces). Those services access the high-performance data store, usually via create, read, update, delete (CRUD) APIs, but may also directly access certain back-end data via API calls or federated views made available by data virtualization.

In many real-life implementations, the API services use the high-performance data store only for queries, while updates are performed directly on the systems of record, according to a CQRS pattern. In more advanced scenarios, the API services can also update the high-performance data store.

- **The integration layer:** This is responsible for keeping the systems of record and the high-performance data store in sync. Synchronization may be unidirectional, when the data store is "read only," or bidirectional, when the data store can also be updated by the API services.

This layer is in charge of asynchronously, notifying events (e.g., "customer XYZ's address has changed") from the systems of record to the data management layer, and possibly vice versa. It is typically implemented using a combination of technologies, such as:

- Extract, transform, load (ETL) — usually for the initial DIH data store loading, but also for periodic refreshes
- Data virtualization
- Event brokering/message-oriented middleware (MOM)
- Change data capture (CDC)
- Application integration platforms — e.g., an enterprise service bus (ESB) or iPaaS
- Stream processing platforms (such as Apache Spark and Flink)

A common combination is CDC to capture events and Apache Kafka to persist events.

The integration layer capabilities along with the API gateway are often provided in the form of a hybrid integration platform (HIP). See "How to Deliver a Truly Hybrid Integration Platform in Steps."

- **Analytics:** Although support for analytics is not a defining characteristic for a DIH, in some situations the data in the data management layer is also utilized for analytic purposes, via a variety of tools (e.g., classic business intelligence tools, data visualization and predictive analytics). In this way, DIH-enabled API platforms can be used to enable augmented transactions scenarios where support for "transactional" APIs is combined with analytics. However, they are not intended to replace analytics-oriented architectures such as data hubs, data warehouses and data lakes (see "Data Hubs, Data Lakes and Data Warehouses: How They Are Different and Why They Are Better Together").
- **The management, monitoring, administration and security capabilities** required to support a business-critical infrastructure.

Note 5: Digital Integration Hub vs. Caching vs. Operational Data Store

- The DIH is an advanced form of caching between the front-end API services layer and the system-of-record layer. At times, the DIH is qualified as "push caching." Notably, data stored in the DIH is "all the relevant data," not — as in classic caching — the "most frequently accessed data," thus allowing extension of the architecture to support additional use cases.

- Some of the design principles and technology characteristics of an operational data store (ODS) are similar to those of the DIH data management layer. However, ODSs, which aggregate transactional data from multiple sources like DIHs, are designed and optimized for operational reporting (see "The Structured Components of the Logical Data Warehouse: Enterprise Warehouse, Mart, Hub and ODS"), whereas DIHs are optimized for API serving. For historical reasons, some organizations refer to their DIH implementation as an ODS.

Note 6: When to Adopt a DIH-Enabled API Platform

Adopt a DIH-enabled API platform when you must tackle projects with some combination of the following requirements:

- Providing a responsive, low-latency and highly available user experience, especially in multiexperience scenarios targeting large constituencies (hundreds of thousands or greater), where the applications require real-time access to systems of record.

- Offloading systems of record from excessive or expensive API-generated workloads.

- Enabling the API services to access data scattered across multiple back-end systems, which would be hard to implement using conventional integration/orchestration techniques.

- Drastically decoupling the front-end API services from the systems of record to favor a dual-speed, bimodal evolution of the two layers aimed at application modernization.

- Maintaining an up-to-date, consolidated picture of fast-changing data that can be used to provide analytical services to support data integration and to detect "business moments."

- Paving the ground for the composable enterprise by exposing system-of-record applications and data in the form of (pseudo) packaged business applications (see "Innovation Insight for Packaged Business Capabilities and Their Role in the Future Composable Enterprise").

Document Revision History

Innovation Insight: The Digital Integration Hub Turbocharges Your API Strategy - 26 June 2018

Recommended by the Authors

Some documents may not be available as part of your current Gartner subscription.

The Applications of the Future Will Be Founded on Democratized, Self-Service Integration

Innovation Insight for Composable Modularity Of Packaged Business Capabilities

Choose the Best Integration Tool for Your Needs Based on the Three Basic Patterns of Integration

Top 3 Trends in Application Architecture That Enable Digital Business

How to Deliver a Truly Hybrid Integration Platform in Steps

The Structured Components of the Logical Data Warehouse: Enterprise Warehouse, Mart, Hub and ODS

Data Hubs, Data Lakes and Data Warehouses: How They Are Different and Why They Are Better Together

Data Fabrics Add Augmented Intelligence to Modernize Your Data Integration

Augmented Data Catalogs: Now an Enterprise Must-Have for Data and Analytics Leaders

Transcend Omnichannel Thinking and Embrace Multiexperience for Improved Customer Experience