MICROSERVICES ARCHITECTURES: WHO OWNS THE DATA?

Give developers control over their own virtual database

TECHNOLOGY OVERVIEW

INTRODUCTION

Microservices architectures speed up innovation in government. Instead of spending months or years building monolithic applications, developers quickly build a suite of small services, each running as an independent process in its own logical machine or Linux® container. As government needs change, developers replace, upgrade, scale, and deploy individual microservices—in far less time than they would need to change the entire application.

But who owns the data in a microservices architecture—the database team or the application team? The answer determines how much agility you will gain from microservices.

In most government organizations, application teams submit schema change requests to the database team. Then they wait while the database team evaluates the request to be sure it will not interfere with performance, security, or resilience. Conflicting priorities create tension: Application teams prize speed while database teams value caution.

The Red Hat® solution meets both teams' goals. With Red Hat JBoss® Data Virtualization, the database team creates reusable virtual databases for each application team. Application teams more quickly introduce new features because they can freely edit the schemas in the virtual database and include them in the continuous integration/continuous delivery (CI/CD) pipeline. Database teams retain complete control over physical datastores. With Red Hat JBoss Data Grid, resilience improves because microservices can continue writing and reading to in-memory data grids even if the central datastore goes down.

IMAGINING THE IDEAL DATA MANAGEMENT SOLUTION FOR MICROSERVICES

Government developer teams want control over their application's data schema and role-based access control (RBAC). But replicating data into dozens or hundreds of databases—one for each application—is not practical. Problems include security risks, high management overhead, and loss of data integrity. Replication erodes the single source of truth.

What if it were possible to give application teams ownership of data without replicating data? In this vision, application teams could:

- Define database schemas without having to own or maintain a physical database.
- Provide role-based access to data.
- Include schemas and changes as part of the CI/CD pipeline.
- Define transactional boundaries, such as percolating application requests across several microservices.

CREATE VIRTUAL DATABASES AND DATA GRIDS WITH RED HAT JBOSS MIDDLEWARE

Government IT teams can safely give data ownership to application teams by using Red Hat JBoss Middleware to create virtual databases and data grids. Table 1 summarizes how this architecture compares to existing government architectures.

TABLE 1. COMPARING GOVERNMENT APPLICATION ARCHITECTURES

APPLICATION ARCHITECTURE	DATABASE ACCESS	DATA OWNERSHIP
Monolithic	Real-time connection between application and central datastore	Database team
Microservices	Real-time connection between microservice and central datastore	Database team—risk of schema changes is higher because changes affect multiple microservices
Microservices plus virtual databases	Each microservice connects to its own virtual database, which is exposed as a representational state transfer (REST) endpoint or Java^{TM} database connectivity (JDBC) service Real-time connection between virtual database and central datastore	Developers—they can freely edit schemas and compositions and then update the mappings between the virtual database and the central datastore

Virtual databases neatly solve the data ownership problem by giving application owners ownership of their data views and access control. But there is one problem: If the underlying database goes down, so do all of the virtual databases.

ADDING RESILIENCE WITH AN IN-MEMORY DATA GRID

Red Hat JBoss Data Grid makes virtual databases resilient. An in-memory, distributed, NoSQL datastore solution, Red JBoss Data Grid synchronizes copies of data across multiple servers (Figure 1). Virtual databases connect in real-time—not to the physical database, but to the distributed data grid. The data grid becomes the source of truth for applications built from microservices.

1. Containerized applications write to the virtual database.
2. The virtual database writes to the data grid, synchronously.
3. The data grid writes to the underlying datastore, asynchronously. To upgrade or migrate the database, the database team simply stops taking asynchronous writes.

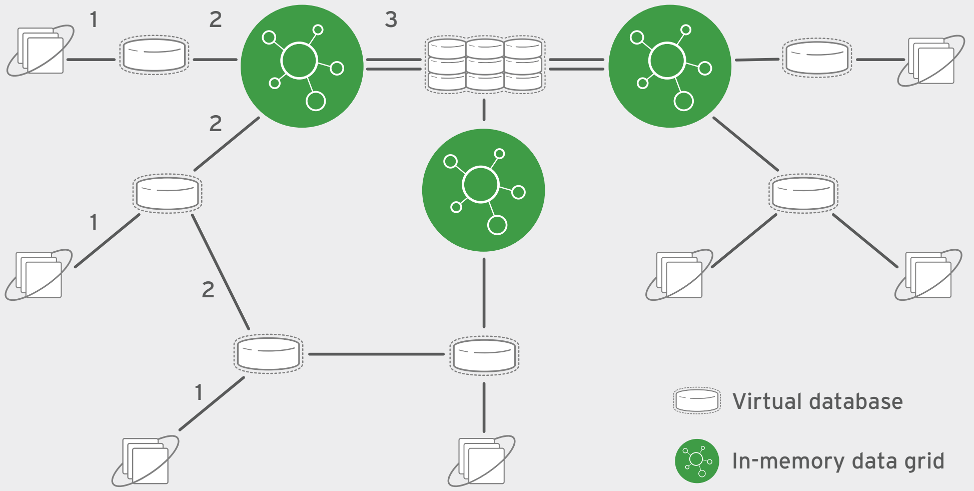

Figure 1. Data layer for microservices

The traditional concern about in-memory activities is resiliency. What happens if a data grid instance goes down? Red Hat JBoss Middleware overcomes this objection by treating the physical datastore as the archive. It is the equivalent of tape in legacy architectures. A data grid node that fails is automatically restored from the central datastore. There is no data loss, no loss of service, and no need to reload or resync the data.

BENEFITS OF VIRTUAL DATABASES AND IN-MEMORY DATA GRIDS

FOR APPLICATION DEVELOPERS

Virtual databases use very few resources, so application teams can create as many as they need. Virtual databases help developers:

- **Innovate faster.** No more waiting for the database team to make schema changes.
- **Simplify deployment.** Developers onboard the database schema to the CI/CD pipeline. They deploy the virtual database right alongside microservices.
- **Reuse data views for new microservices.** Developers can reuse and combine existing data views for new microservices.

DATA MANAGEMENT TEAM

By giving application teams ownership of their data, government data management teams:

- **Reclaim time.** No need to evaluate and fulfill requests for schema changes.
- **Increase resiliency.** A data grid node that goes down is automatically restored from the central datastore.
- **Improve control over data security.** The database team centrally controls access to underlying data sources by user or role. No need to individually configure each data source for access by dozens of microservices.

HOW RED HAT JBOSS DATA VIRTUALIZATION WORKS

Figure 2 shows how data flows between sources and consumers. This illustrates Data-as-a-Service (DaaS), with three functional layers:

- **Connect.** The virtual database establishes real-time connections to any data source. Teeid, a Red Hat JBoss project, distributes application queries across all underlying sources.
- **Compose.** Teams compose virtual tables to create the ideal application view.
- **Consume.** Microservices provision data via any interface, including JDBC, open database connectivity (ODBC), REST, open data protocol (OData), simple object access protocol (SOAP), and others.

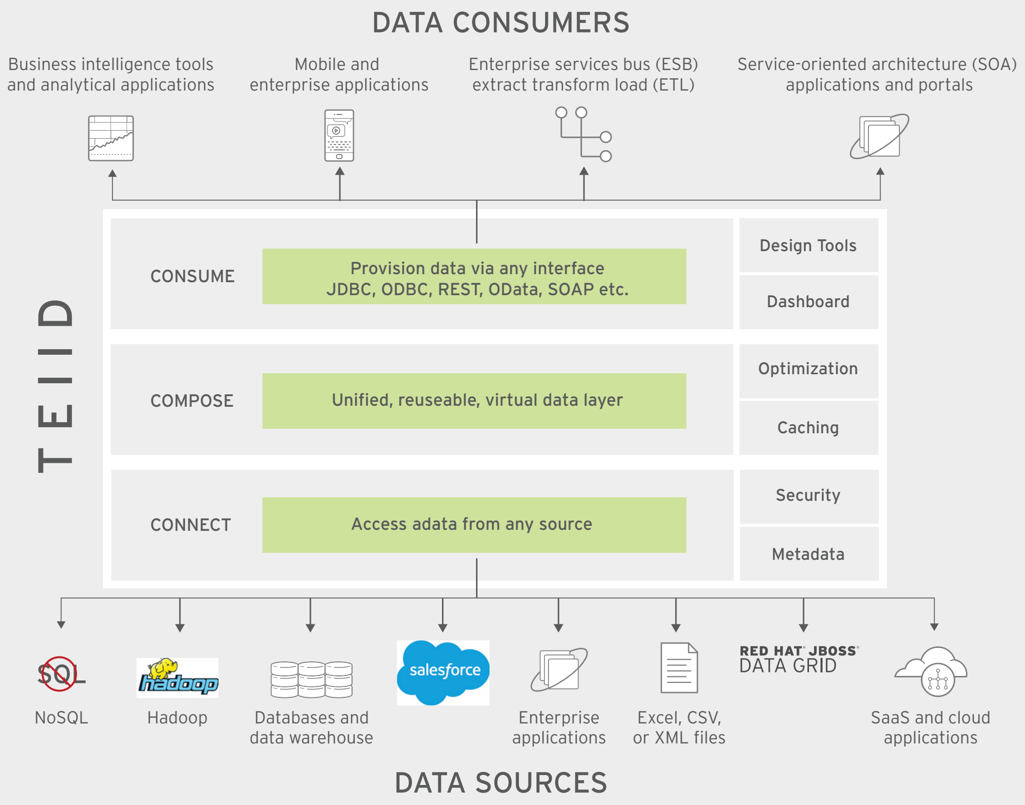
DATA CONSUMERS
Business intelligence tools and analytical applications
Mobile and enterprise applications
Enterprise services bus (ESB) extract transform load (ETL)
Service-oriented architecture (SOA) applications and portals
TEIID
CONSUME
Provision data via any interface JDBC, ODBC, REST, OData, SOAP etc.
COMPOSE
Unified, reuseable, virtual data layer
CONNECT
Access adata from any source
Design Tools
Dashboard
Optimization
Caching
Security
Metadata
SQL
NoSQL
hadoop
Hadoop
Databases and data warehouse
salesforce
Enterprise applications
Excel, CSV, or XML files
RED HAT JBOSS DATA GRID
SaaS and cloud applications
DATA SOURCES

Figure 2. Red Hat JBoss Middleware approach to virtual databases and data grids

CONCLUSION

The agility you gain from microservices architectures depends in large part on who owns the data. Many more entities access the datastore, so expecting the database team to quickly fulfill requests for schema changes is unrealistic.

Red Hat JBoss Middleware relieves the database team by creating a virtual data layer owned by application teams. Developers can create the ideal data view for their microservices and include the view in the CI/CD pipeline. Freed from making schema changes, database teams reclaim time to focus on security, performance, and data integrity.

There is more than one way to manage data in a microservices architecture. The Red Hat approach offers important advantages:

- Simpler operations—the result of completely decoupling the application development and database life cycles.
- Avoidance of data replication—strengthening data integrity and security while reducing management overhead.

Learn more about Red Hat JBoss Data Virtualization at redhat.com/en/technologies/jboss-middleware/data-virtualization.

Learn more about Red Hat JBoss Data Grid at redhat.com/en/technologies/jboss-middleware/data-grid.

ABOUT RED HAT

Red Hat is the world's leading provider of open source software solutions, using a community-powered approach to provide reliable and high-performing cloud, Linux, middleware, storage, and virtualization technologies. Red Hat also offers award-winning support, training, and consulting services. As a connective hub in a global network of enterprises, partners, and open source communities, Red Hat helps create relevant, innovative technologies that liberate resources for growth and prepare customers for the future of IT.

facebook.com/redhatinc
@redhatnews
linkedin.com/company/red-hat

NORTH AMERICA	EUROPE, MIDDLE EAST, AND AFRICA	ASIA PACIFIC	LATIN AMERICA
1 888 REDHAT1	00800 7334 2835	+65 6490 4200	+54 11 4329 7300
	europe@redhat.com	apac@redhat.com	info-latam@redhat.com