Making Mainframe Data Available for the Entire Organization with Data Virtualization

A Whitepaper

Rick F. van der Lans
Independent Business Intelligence Analyst
R20/Consultancy

July 2014

Sponsored by

Table of Contents

1 Management Summary

Unlocking Mainframe Data – Organizations running mainframes still collect massive amounts of data on these machines. Over the years, they have invested heavily in development of these systems that form their IT backbone. In fact, countless organizations still depend on them.

Initially, most of the mainframe databases were developed to support one or two transactional applications. However, mainframe data can be used for other business purposes as well, such as cross-database reporting, discovering long term trends and patterns, supporting operational analytics, Web and Mobile applications.

To make mainframe data available for these other use cases can be a technological challenge due to the products and technology used to store the data. A lot of mainframe data is not stored in SQL database servers, but in so-called *pre-SQL database servers*, such as CA IDMS, IBM IMS, and Software AG Adabas, and in files systems such as VSAM. These products use non-SQL concepts and don't support SQL itself.

Data Virtualization – The on-demand integration and transformation features of a *data virtualization server* make it possible for users to access mainframe data as easy as any other data source. For example, through data virtualization, mainframe data stored in an IMS and an Adabas database can be integrated with data stored in an Oracle database running on a Unix server. Thus, data virtualization allows for easy and agile unlocking of all the mainframe data for any use case, including reporting, analysis, web and mobile applications. It unlocks the real value of mainframe data to the entire enterprise.

Purposes of the Whitepaper – This whitepaper has two purposes. One, it describes the importance of data virtualization to make mainframe data available to the entire organization. Second, it lists the requirements that a data virtualization must support to unlock mainframe data efficiently.

2 Unlocking Mainframe Data

The Priceless Value of Mainframe Data – Numerous organizations still manage their most critical enterprise data with mainframe-based IT systems. Without these systems, many of them would be lost. Despite all the hype around big data and the introduction of new data storage technologies and platforms, enterprise data stored and managed on mainframes is still crucial and priceless.

For over more than twenty years, organizations have invested heavily in developing mainframe-based systems. Most of them were developed to support and streamline business processes. For example, financial organizations developed systems to service loans, process cash deposits and withdrawals, and process payments and cheques. These systems form the backbone of most retail banks. They collect the most essential data that a bank can own. Their dependency on them is so high, that if all the mainframes on this planet would magically disappear overnight, the entire financial world would grind to a halt in hours. Likewise, oil companies, retailers, government, and healthcare, they all have major investments in these systems and are dependent on them.

Mainframe Data is Hidden in Transactional Applications – Initially, most of the mainframe databases were developed to support one or two transactional applications and still do. In other words, this valuable data is used for a limited number of use cases. This is quite unfortunate, because mainframe data can be used for a wide range of use cases. In other words, it can be used by a much wider audience, for multiple business purposes, and for a more diverse set of use cases. Here are some examples of use cases for which mainframe data can be used:

- For cross-database reporting to present, for example, a 360 degrees view of customers to support call center applications.
- To present key performance indicators in dashboards to show, for example, manufacturing processes spill levels or call center productivity.
- For analytics to discover long term trends and patterns, such as customer buying patterns, or to target customers better in marketing campaigns.
- To support operational analytics where data is investigated for, for example, fraudulent banking activities.
- To augment a Master Data Management system with up-to-date data.
- To support non-mainframe-based applications that need real-time access to mainframe data, such as Web and mobile applications.
- To enrich non-mainframe data, such as big data containing weblog records or sensor data coming from factory machines.

In general, every time when mainframe data is used for another than its initial purpose, its value to the organization increases.

The Technological Challenges of Unlocking Mainframe Data – The business value of making mainframe data available for other purposes is clear, however, it's not technically easy. Transactional systems themselves have no problems inserting and accessing mainframe data, because they were developed hand in hand with the mainframe databases. But unlocking mainframe data for other uses case is a major technological challenge. The key reason is that most mainframe data is not stored in SQL database servers, but in so-called *pre-SQL database servers*, such as CA IDMS, IBM IMS, and Software AG Adabas, and also in file systems such as VSAM. These products don't store data in classic tables consisting of columns and records, but use completely different sets of concepts, such as logical parents, logical children, owners, and members. These concepts were designed specifically for transactional purposes, not for anything else.

The challenge is to make all that mainframe data available for reporting, historical analysis, data mining, web and mobile applications, and so on. But how? Most tools for reporting, analytics, and development expect that data is stored in tables and that they can use SQL to query it.

Is Copying Mainframe Data the Solution? – The classic solution to make mainframe data available is to copy it from these pre-SQL databases to a SQL database. In this case, an ETL product or ETL-based solution is used to periodically extract, transform, and integrate the mainframe data with data from other systems. Subsequently, it's loaded in a dedicated database (usually residing on another platform than the mainframe), and from there it's copied to, for example, a data warehouse to make it available for all forms of reporting and analytics. This approach has four drawbacks:

- **No Real-Time Data:** With this approach, the applications don't see live data but data that can be one hour, one day, or maybe even one week old. This may be acceptable for long term trend analysis, but not for all applications, such as dashboards, and Web and mobile applications. They need to operate with real-time data, not with yesterday's data.

- **Read-Only Data:** With ETL, copied data can only be read. Some applications want to be able to change and insert new data.

- **Complex Architecture:** Copying data requires management, monitoring, maintenance, and optimization of extra databases and requires the scheduling of ETL jobs. This may all lead to a complex and costly architecture.

- **Data Security:** Mainframe data that has been copied falls outside the control of the existing security mechanisms. Additional ones must be implemented to guarantee safe storage and safe transmission of the mainframe data. This is particularly relevant for financial services, government, and healthcare due to regulatory requirements.

3 What is Data Virtualization?

Data Integration with Data Virtualization – The last few years a new data integration technology called *data virtualization* has been making headway in the data integration market. According to Gartner[1], of all the data integration technologies, it's the only one for which investment is growing. With data virtualization, all the data, regardless of where it's store, how it's stored, and how it must be accessed, can be presented as one large, logical database that can be accessed using one database language.

Data Virtualization in a Nutshell – Data virtualization is a technology for integrating, transforming, and manipulating data that's available in all kinds of data sources and for presenting all that data as one logical database. When data virtualization is applied, an abstraction and encapsulation layer is provided that, for applications, hides most of the technical aspects of how and where data is stored; see Figure 1. Because of this layer, applications don't need to know where all the data is physically stored, whether it's on a mainframe or not, how the data should be integrated, where the database servers run, how to insert and update the data, what the required APIs are, which database language to use, and so on. When data virtualization is deployed, to every application it feels as if one large database is accessed.

Applications can use different interfaces to access data, including SQL, SOAP, and REST. When, for example, SQL is used to access data stored with IBM IMS, the data virtualization server transforms SQL into IMS DL/I. Data virtualization servers can also change the structure of the source data, if necessary. If data comes from multiple systems, the data virtualization server integrates the data and presents a unified result to the application.

Definition of data virtualization[2]: *Data virtualization is the technology that offers data consumers a unified, abstracted, and encapsulated view for querying and manipulating data stored in a heterogeneous set of data stores.*

[1] Ted Friedman (Gartner), *Modernize Your Data Integration Capabilities for Diverse Use-Cases*, 2014.
[2] Rick F. van der Lans, *Data Virtualization for Business Intelligence Systems*, Morgan Kaufmann, 2012.

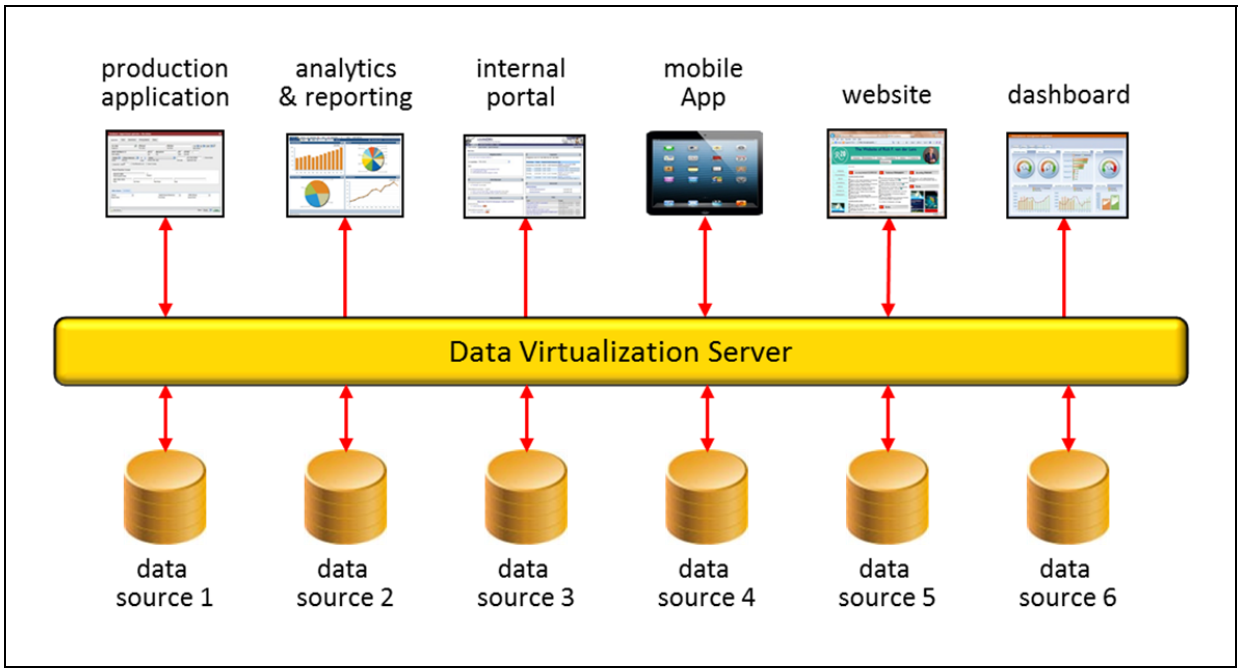

Figure 1 *Data virtualization servers make a heterogeneous set of data sources look like one logical database to the applications. The data virtualization server is responsible for data integration.*

The Key Advantages of Data Virtualization – The key advantages of data virtualization when deployed to retrieve and integrate mainframe data are the following:

- **Easy Data Access:** Some mainframe database servers support complex concepts and offer only highly technical interfaces. This makes the data in such systems difficult to access. With data virtualization servers, users and applications can use easier-to-use languages, such as SQL, to access data. All the technical concepts and interfaces are hidden.

- **On-Demand Data Transformation:** When users retrieve data via a data virtualization server, that data is retrieved from the source systems and transformed to the right SQL table structures *live*. So, when data is accessed from a mainframe data source, it's not first copied to and stored in another database or file, but is streamed live to the users instead. Therefore, data virtualization supports on-demand data transformation.

- **On-Demand Data Integration:** In contrast to ETL tools, data virtualization servers integrate data available from different sources live. As with on-demand data transformation, there is no need to store the integrated result first before it can be used.

- **Real-Time Data:** Because the mainframe data is accessed directly and not a database containing a copy of the data, applications see real-time data. Users see the current state of the data and not yesterday's data.

- **Read/Write Interface:** With data virtualization, mainframe data is accessed directly. This allows applications to read the data as well as insert and update data.

- **Increased Agility:** Data virtualization increases agility, because developers don't have to use the complex languages of the mainframe database servers. In addition, because the applications have been decoupled from the data sources, it's easier to make changes to the source systems without impacting the applications. All this dramatically improves the agility of an IT system.

- **Improved Time-to-market:** If data is easier to access, integrating existing data sources and plugging in new data sources is less time-consuming. In other words, it improves the time-to-market—organizations can react faster.

- **Simple Architecture:** Because data virtualization deploys on-demand data integration, no additional databases have to be managed and maintained.

- **Data Security:** Because on-demand integration is deployed, all the security mechanisms protecting the mainframe data still apply.

4 Requirements for Mainframe Access using Data Virtualization

This section lists the essential requirements for data virtualization servers when accessing data stored in mainframe databases.

Efficient Database Access – Because the concepts supported by most mainframe database servers are not based on the simple table-column-record concept, translating SQL to their database languages is a complex process. It's not a simple 1:1 conversion of concepts. In addition, translating SQL to a non-SQL program is not good enough, it must be a highly efficient program. A data virtualization server should also be able to exploit the performance-improving features supported by these mainframe products.

Minimal Interference – Minimal interference entails two aspects: minimal interference on operational application processing and minimal mainframe resource consumption. Most mainframe database servers support an intense transactional workload. Under no condition should the extra workload generated by the data virtualization server interfere with this ongoing transactional workload, it must not destabilize the performance, and care must be taken with respect to technical concurrency aspects such as locking.

In addition, data virtualization servers consume CPU resources: SQL statements must be translated, security rules must be checked, extra data processing must be performed, and so on. All this processing leads to resource consumption and may indirectly lead to interference on the transactional applications. Data virtualization servers should minimize this form of interference.

Integration on the Mainframe – Many data integration tools that can access mainframe data, don't run on the mainframe itself. If multiple data sources residing on the mainframe must be integrated, they retrieve all the relevant data from all these sources and execute the integration outside the mainframe. This can lead to inefficient data transmission between the mainframe and the other platform. For example, the result of a join of two mainframe sources may be a small amount of data, but the data needed to generate this small result may be considerable. In this case, it's much more efficient to "push" the join to the mainframe and ship a minimal amount of data to the data virtualization server running externally. In short, integration of mainframe data sources is much more efficient if it's executed on the mainframe, and only

the final result is returned to the application running on another machine. Note also that the integration of data sources on the mainframe should not lead to interference of the operational applications.

In addition, to integrate mainframe data with off-mainframe data, if the majority of it resides off-host, a data virtualization server must be able to move the data to the mainframe where it executes the integration process. Enabling integration off-host must be done via the most cost-effective approach.

Use Existing Data Security Rules – For many mainframe database servers specific security technology has been developed and specific security rules apply. For example, they come with their own security systems to define which users are allowed to do what with the data. Data virtualization servers must be able to adopt these security rules and definitely not circumvent them.

Classic Data Virtualization Requirements – Besides the above-mentioned requirements that are specific to accessing mainframe-based data sources, data virtualization servers should also support the more classic requirements, such as:

- Access to large heterogeneous sets of data store technologies and systems
- Caching functionality to store and freeze results temporarily
- Heterogeneous distributed transactions
- Join optimization techniques
- Nesting of virtual tables
- Single sign-on

For a more extensive list of requirements, see this article[3].

5 Specifications of Rocket Data Virtualization

The History of Rocket Data Virtualization – Development of *Rocket Data Virtualization Server* (Rocket DVS) started in 2013. The product works from a runtime engine that is unique in its ability to exploit the IBM System z Integrated Information Processor (zIIP) to reduce mainframe MIPS capacity usage related to integration. From the first day of development, Rocket DVS was meant to simplify access to data no matter where it resides. For the engineers at Rocket, the mainframe was the logical first step in data virtualization.

Heterogeneous Data Access – Rocket DVS can access a wide range of mainframe database servers, such as CA IDMS, IBM CICS, DB2 z/OS, IMS TM & DB, Software AG Adabas and Natural. In addition, typical mainframe file systems, such as Sequental files and VSAM files, can be accessed. The product has taken advantage of IBM's data integration standard DRDA (Distributed Relational Database Architecture). Through DRDA it can access many SQL database servers, including IBM DB2, Informix and PureData (Netezza); Oracle; Microsoft SQL Server, and Apache Derby. Because of DRDA, access to these database servers by Rocket DVS is not limited to read-only access. Data can be inserted, updated, and deleted. Even distributed transactions are supported. The product itself can operate as a DRDA source as well.

[3] Rick F. van der Lans, *The Requirements of the Data Delivery Platform*, February 2010, see http://www.b-eye-network.com/view/12582

Supported Interfaces – Client applications access Rocket DVS through a variety of interfaces, including SQL (JDBC, ODBC), NoSQL (JSON/BSON), Services (SOAP, REST), Web (HTTP/HTTPs), and Events (MQ, XML).

The Architecture of Rocket DVS – The product runs on the mainframe. To be precise, it runs on the *zIIP processor* (System Z Integrated Information Processor). Like any computer, the mainframe deploys one or more general purpose processors. But since 2006, the IBM Systems z9 (and up) mainframes can be extended with special purpose processors called zIIP. These are designed to relieve the mainframe's central processors. The idea is based on older special purpose processors, such as the zAAP and IFL, which were both designed for offloading Java and Linux processing. Applications running on zIIP processors don't interfere with mission critical enterprise applications running on the general purpose central processors. For example, IBM DB2 for z/OS makes use of zIIP for certain tasks. Currently, the zIIP is available on all IBM zEnterprise, System z10, and System z9 servers. For more information on zIIP, see [4].

Because Rocket DVS runs on zIIP processors, all its processing does not interfere with the applications. Evidently, when it accesses mainframe databases processing is executed on the central processors. But all the processing to integrate data sources, transform data, and generate the correct statements, is all done on the zIIP processor. In fact, Rocket Software claims that 99% of the integration processing done by Rocket DVS, such as SQL to non-relational data queries and Web services/SOA workloads, is zIIP eligible and doesn't run on the central processors.

Meeting the Requirements – Section 4 lists critical requirements for data virtualization servers to access mainframe data. Here we describe how Rocket DVS supports them.

- **Efficient Database Access:** Rocket DVS has been optimized to access mainframe database servers using SQL.

- **Minimal Interference:** Through the years a vast amount of knowledge has been built-up by the team that has worked on Rocket DVS. Currently, the programs generated to access the databases are as efficient as can be, although it will always be an extra workload. Rocket DVS minimizes resource consumption on the mainframe by running on the zIIP processors. So, this form of interference is negligible, because the applications and Rocket DVS are not competing for processing resources on the central processors.

- **Integration on the Mainframe:** Rocket DVS runs on the mainframe, so if multiple data sources on the mainframe must be integrated, the integration takes place on the mainframe itself minimizing the amount of data transmitted to applications not hosted on the mainframe. It can also move off-mainframe data to the mainframe for integration needs.

- **Use Existing Data Security Rules:** Rocket DVS supports all the mainframe security protocols including RACF, ACF2, and TopSecret. These popular mainframe security systems are not bypassed.

- **Classic Data Virtualization Requirements:** Rocket DVS supports the classic features supported by competing data virtualization servers.

[4] See http://www.ibmsystemsmag.com/mainframe/administrator/performance/add_some_zIIP_to_your_mainframe/

Summary – A mainframe resident data virtualization server such as Rocket DVS is suited for environments in which the majority of the enterprise data is stored on the mainframe. Rocket DVS resides on the mainframe and because it runs on special purpose processors, it minimizes interference on mission critical enterprise applications running on the general purpose central processors. Rocket DVS also makes use of the popular mainframe security systems.

Rocket DVS can be used to unlock mainframe data and to make the data available for other use cases, such as analysis, interactive reporting, data mining, web and mobile applications. In other words, Rocket DVS helps to unlock the real value of mainframe data to the entire enterprise.

In an overall data virtualization architecture, there is a place for specialized data virtualization servers such as Rocket DVS that address the unique requirements of the mainframe and that can also work in combination with other data virtualization products.

About the Author Rick F. van der Lans

Rick F. van der Lans is an independent analyst, consultant, author, and lecturer specializing in data warehousing, business intelligence, database technology, and data virtualization. He works for R20/Consultancy (www.r20.nl), a consultancy company he founded in 1987.

Rick is chairman of the annual European Enterprise Data and Business Intelligence Conference (organized annually in London). He writes for the eminent B-eye-Network.com[5] and other websites. He introduced the business intelligence architecture called the *Data Delivery Platform* in 2009 in a number of articles[6] all published at BeyeNetwork.com. The Data Delivery Platform is an architecture based on data virtualization.

He has written several books on SQL. Published in 1987, his popular *Introduction to SQL*[7] was the first English book on the market devoted entirely to SQL. After more than twenty years, this book is still being sold, and has been translated in several languages, including Chinese, German, and Italian. His latest book[8] *Data Virtualization for Business Intelligence Systems* was published in 2012.

For more information please visit www.r20.nl, or email to rick@r20.nl. You can also get in touch with him via LinkedIn and via Twitter @Rick_vanderlans.

About Rocket Software, Inc.

Rocket Software is a global software development firm that builds enterprise products and delivers enterprise solutions in the following segments: Business Information and Analytics; Storage, Networks, and Compliance; Application Development, Integration, and Modernization; and Database Servers and Tools. Rocket is engaged in business and technology partnerships with IBM, EMC, Fujitsu, HP Enterprise Services, Hitachi Data Systems, Avaya, Epicor, and many others. The company is headquartered in Waltham, Massachusetts, USA. For more information, visit www.rocketsoftware.com or follow them on Twitter @rocket.

[5] See http://www.b-eye-network.com/channels/5087/articles/
[6] See http://www.b-eye-network.com/channels/5087/view/12495
[7] R.F. van der Lans, *Introduction to SQL; Mastering the Relational Database Language*, fourth edition, Addison-Wesley, 2007.
[8] R.F. van der Lans, *Data Virtualization for Business Intelligence Systems*, Morgan Kaufmann Publishers, 2012.