Redbooks
ibm.com/redbooks

Apache Spark for the Enterprise
Setting the Business Free

Oliver Draese

Eberhard Hechler

Hong Min

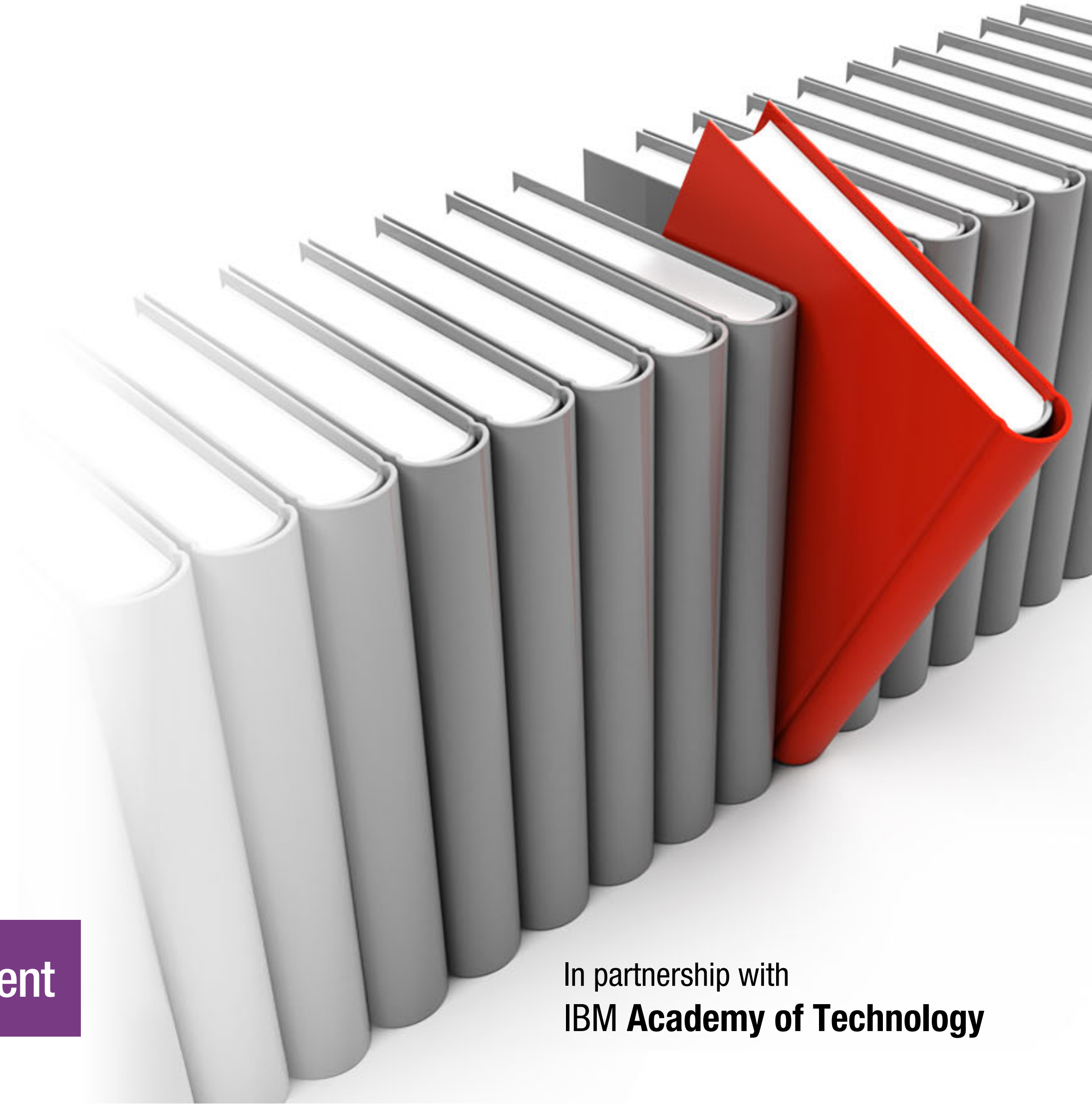
Catherine Moxey

Pallavi Priyadarshini

Mark Simmonds

Mythili Venkatakrishnan

George Wang



Information Management

In partnership with
IBM Academy of Technology

IBM®

Redpaper

International Technical Support Organization

Apache Spark for the Enterprise: Setting the Business Free

February 2016

REDP-5336-00

Note: Before using this information and the product it supports, read the information in “Notices” on page v.

First Edition (February 2016)

This edition applies to Apache Spark on IBM z Systems platforms.

This document was created or updated on February 9, 2016.

Contents

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive, MD-NC119, Armonk, NY 10504-1785, US

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided “AS IS”, without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at “Copyright and trademark information” at http://www.ibm.com/legal/copytrade.shtml

The following terms are trademarks or registered trademarks of International Business Machines Corporation, and might also be trademarks or registered trademarks in other countries.

BigInsights®
Bluemix®
CICS®
Cloudant®
Cognos®
DB2®
DB2 Connect™
Guardium®
IBM®
IBM PureData®
IBM Watson™
IBM z Systems™
IBM z13™
IMS™
InfoSphere®
MVS™
Parallel Sysplex®
PureData®
QMF™
Redbooks®
Redpaper™
Redpapers™
Redbooks (logo) ®
SPSS®
Tivoli®
Unica®
WebSphere®
z Systems™
z/OS®
z13™

The following terms are trademarks of other companies:

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Java, and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Other company, product, or service names may be trademarks or service marks of others.

Find and read thousands of IBM Redbooks publications

- Search, bookmark, save and organize favorites
- Get personalized notifications of new content
- Link to the latest Redbooks blogs and videos

Get the latest version of the Redbooks Mobile App

iOS **Download Now** Android

Promote your business in an IBM Redbooks publication

Place a Sponsorship Promotion in an IBM® Redbooks® publication, featuring your business or solution with a link to your web site.

Qualified IBM Business Partners may place a full page promotion in the most popular Redbooks publications. Imagine the power of being seen by users who download millions of Redbooks publications each year!

ibm.com/Redbooks
About Redbooks → Business Partner Programs

THIS PAGE INTENTIONALLY LEFT BLANK

Preface

Analytics is increasingly an integral part of day-to-day operations at today's leading businesses, and transformation is also occurring through huge growth in mobile and digital channels. Enterprise organizations are attempting to leverage analytics in new ways and transition existing analytics capabilities to respond with more flexibility while making the most efficient use of highly valuable data science skills. The recent growth and adoption of Apache Spark as an analytics framework and platform is very timely and helps meet these challenging demands.

The Apache Spark environment on IBM z/OS® and Linux on IBM z Systems™ platforms allows this analytics framework to run on the same enterprise platform as the originating sources of data and transactions that feed it. If most of the data that will be used for Apache Spark analytics, or the most sensitive or quickly changing data is originating on z/OS, then an Apache Spark z/OS based environment will be the optimal choice for performance, security, and governance.

This IBM® Redpaper™ publication explores the enterprise analytics market, use of Apache Spark on IBM z Systems™ platforms, integration between Apache Spark and other enterprise data sources, and case studies and examples of what can be achieved with Apache Spark in enterprise environments. It is of interest to data scientists, data engineers, enterprise architects, or anybody looking to better understand how to combine an analytics framework and platform on enterprise systems.

Authors

This paper was produced by a team of specialists from around the world.

Oliver Draese is a Senior Technical Staff Member within the IBM Silicon Valley Lab. He has 15 years of experience with IBM DB2® and his expertise includes database engine design, accelerators, and advanced analytics.

Eberhard Hechler is an Executive Architect from the IBM Germany R&D Lab. He is a member of DB2 Analytics Accelerator development. In his 2.5 years at the IBM Kingston Lab in New York, Eberhard has worked in software development and performance optimization. He then worked with IBM DB2 for MVS™, Master Data Management, and Hadoop and Spark integration. He has worked worldwide with IBM clients from various industries on a vast number of topics, such as DWH and BI, information architectures, and industry solutions. During 2011 - 2014, he was at IBM Singapore, working as the Lead Big Data Architect in the Communications Sector of IBM Software Group (SWG). He is a member of the IBM Academy of Technology Leadership Team and co-authored the following books: *Enterprise Master Data Management*, Pearson, 2008, ISBN: 0132366258; *The Art of Enterprise Information Architecture*, Pearson, 2010, ISBN: 0137035713; *Beyond Big Data*, Pearson, 2014, ISBN: 013350980X.

Hong Min is a Senior Technical Staff Member and a Master Inventor at the IBM T. J. Watson Research Center, US. She joined IBM in 1997 and her current research interests include database systems, data processing system integration, and acceleration technologies for data processing. Hong has co-authored several IBM Redbooks® and Redpaper publications, and has published multiple research papers.

Catherine Moxey is an IBM Senior Technical Staff Member in CICS® technical strategy and design, based at IBM Hursley in the UK. Catherine has over 30 years of experience as a Software Engineer, 26 of those with IBM. Her areas of focus include CICS performance and optimization, event processing support, and analytics.

Pallavi Priyadarshini is a Senior Technical Staff Member at Spark Technology Center, IBM Analytics, working at the India Labs. Pallavi has led global teams over the last 14 years in delivering mission-critical data-centric solutions for enterprise customers. Currently, she leads the integration of Apache Spark with Databases and Spark customer enablement. Prior to her current role, Pallavi was the Architect and Product Manager of the IBM DB2 Connect™ portfolio, consisting of Java, C, and open source drivers for on-premises and cloud databases. As part of the DB2 z/OS development team in Silicon Valley Labs, she has developed key server features for application modernization. Pallavi is a regular speaker at global conferences. She has authored several patents and publications in databases and tools. She has completed her Master in Computer Science degree from San Jose State University, California, and Bachelor in Computer Science degree from Nanyang Technological University, Singapore.

Mark Simmonds is Program Director for IBM cognitive analytics in z Systems. He is a recognized thought leader and speaker on Spark, big data, and analytics. He has 20 years of IBM service and spent three years as an IBM Systems Architect responsible for infrastructure design and corporate technical architecture in large financial institutions. Prior to joining IBM, he was head of Information and IT for the National Health Service (UK). He has a number of author recognition awards, written articles for technical and business journals, and is a professional member of the British Computer Society.

Mythili Venkatakrishnan is an IBM Senior Technical Staff Member and is the z Systems Architecture and Technology Lead for Analytics. Mythili has been with IBM for over 25 years, all in the mainframe environment working with clients in various capacities. Her focus areas have been diverse and include: analytics, industry solutions, business resilience architecture and design, availability management, systems design, and solution prototyping. Most recently, Mythili has been working with z Systems clients, IBM colleagues, IBM Business Partners and the broader ecosystem to enable the integration of analytics with transaction environments with a focus on Apache Spark for z Systems.

George Wang is a Software Engineer with IBM DB2 for z/OS Development at the Silicon Valley Lab of California, US. He is the technical advocate and liaison for large banking customers on z Systems. He is the Chair of DB2 Design Review Board overseeing the review and approval of all DB2 technical designs. His technical expertise focuses on architecture design of core system engine components for developing high availability features and providing solutions for warehouse applications in support of high-volume online transactional database processing. He engages on multiple projects for cutting edge technologies to incorporate advanced in-memory analytics with big data support on DB2 for z/OS.

Thanks to the following people for their contributions to this project:

- Martin Keen, IBM Technical Content Services Project Leader
- Andy Armstrong, IBM CICS Early Adoption Lead
- Betty Patterson, IBM Distinguished Engineer, IMS™ Chief Architect

Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our papers to be as helpful as possible. Send us your comments about this paper or other IBM Redbooks publications in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:

  ibm.com/redbooks

- Send your comments in an email to:

  redbooks@us.ibm.com

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400

Stay connected to IBM Redbooks

- Find us on Facebook:

  http://www.facebook.com/IBMRedbooks

- Follow us on Twitter:

  http://twitter.com/ibmredbooks

- Look for us on LinkedIn:

  http://www.linkedin.com/groups?home=&gid=2130806

- Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:

  https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm

- Stay current on recent Redbooks publications with RSS Feeds:

  http://www.redbooks.ibm.com/rss.html

1

Analytics market overview

No matter which way you look at it, analytics is providing significant value by offering deeper insight into what is happening in the business, why it is happening, predicting what might happen next, and taking the appropriate course of action. This can help lower business risks, lower costs, increase opportunities for growth, and help provide smarter, more confident and accurate decisions. However, analytics is only as good as the data it consumes in terms of its quality, completeness, and consistency. Not only must analytics solutions be agile in terms of being able to respond across a broad time spectrum ranging from hours to milliseconds, but data scientists must be able to interact with analytics solutions and complex data sets and also construct ad hoc analytics applications quickly and efficiently.

This chapter addresses the following topics:

- Analytics opportunities and challenges
- IBM z Systems platform: A unique analytics implementation

1.1 Analytics opportunities and challenges

A number of trends today are driving unprecedented use of analytics within organizations. For example, 44% of CIOs reported they will spend on business analytics.[1] And 80 - 90% of companies indicate that big data analytics is either their top priority or in the top three.[2] Nearly 9 of 10 leading companies are gaining major competitive advantage from big data and analytics, compared to 4 of 10 lagging companies.[3]

The return on investment (ROI) is significant. For every $1 US (dollar) invested in analytics, $13 is the ROI.[4] Predictive analytics has been demonstrated to provide a two hundred and fifty percent ROI.[5]

Whether measuring success on revenue, business outcome or competitive advantage front-runners using analytics significantly out perform their competitors.[6]

Winners in their respective industries are often organizations that infuse analytics everywhere. Instead of confining analytics to a few individuals or a single department, they make analytics easily available to business users throughout the organization, and also to data scientists and other data experts. Instead of making analytics a mysterious process, they infuse it into business processes.

As a result, they are able to move quickly from insight to positive business outcomes, in a whole range of areas, from attracting new customers to managing risk to prioritizing investment for innovation.

1.1.1 Bigger and bigger data

By contrast, not all organizations, however, are well positioned to leverage the full potential of analytics. Big data is creating new opportunities to glean actionable insights from new varieties of data and content coming at organizations in huge volumes and at accelerated velocity. Based on a recent IBM study, 65% of businesses are not yet using big data for business advantage.[7] The facts about big data are interesting; for example, when people hear the term *big data*, many people think of social media, email, and unstructured data. However, many analysts show that the top two data types used in big data initiatives are transactional data and log data.

1.1.2 Consumer expectations

The demand for quality customer interactions is on the rise because consumers now have a multitude of choices related to who they do business with and how they choose to interact with each vendor. Winning in today's market requires competitive product offerings and also demands superior customer interactions in order to attract, maintain, and grow the client base and ultimately improve business performance. Creating a positive customer experience is critical because it has been proven to have a direct impact on the success or failure of a business. Customer service organizations must be equipped with analytics to drive a high impact customer-focused organization. A growing consumer demand exists for real-time

[1] Morgan Stanley CIO Survey, April 2015

[2] General Electric, November 2014

[3] IBM Center for Applied Insights, November 2014

[4] Nucleus Research, Report O204, September 2014

[5] Nucleus Research, March 2014 (248% ROI based on Mueller Fabrication case study)

[6] IBM Institute for Business Value, 2014 Study on Analytics

[7] IBM Institute for Business Value, 2015

transactions, driven partly by the ubiquity of smartphones and other connected devices, which have catalyzed consumer expectation for immediacy.[8]

1.1.3 Mobile and analytics collide

A double revolution is happening on the analytics and mobile axes (Figure 1-1). Data is stored and interacted with in various ways, such as these:

- *System of record*: Highly structured, tabulate, rich in value, and often transactional
- *System of engagement*: Differently structured, sparse, ambiguous, social media, email, file shares, and collaborative solutions
- *System of insight*: Data reservoirs, logical data warehouses, Hadoop clusters, and more

Organizations are moving toward becoming the *insightful enterprise*. In parallel, the technology by which users and customers interface with data has evolved rapidly (from green screen terminals and PCs to smart connect devices and wearable technology); it is moving toward what is known as the *situational enterprise*. Combine these two paradigms together and ultimately the market is heading toward what is being referred to as the *individual enterprise*. Organizations are trying to build deep digital personal relationships with each and every one of us. Each one of us is an enterprise. Now we see very targeted personal marketing, personal sales, and personal service. Mobile also adds two more dimensions to analytics (geographic location and time sensitivity) pushing analytics closer to real time and to act in the *mobile moment*.

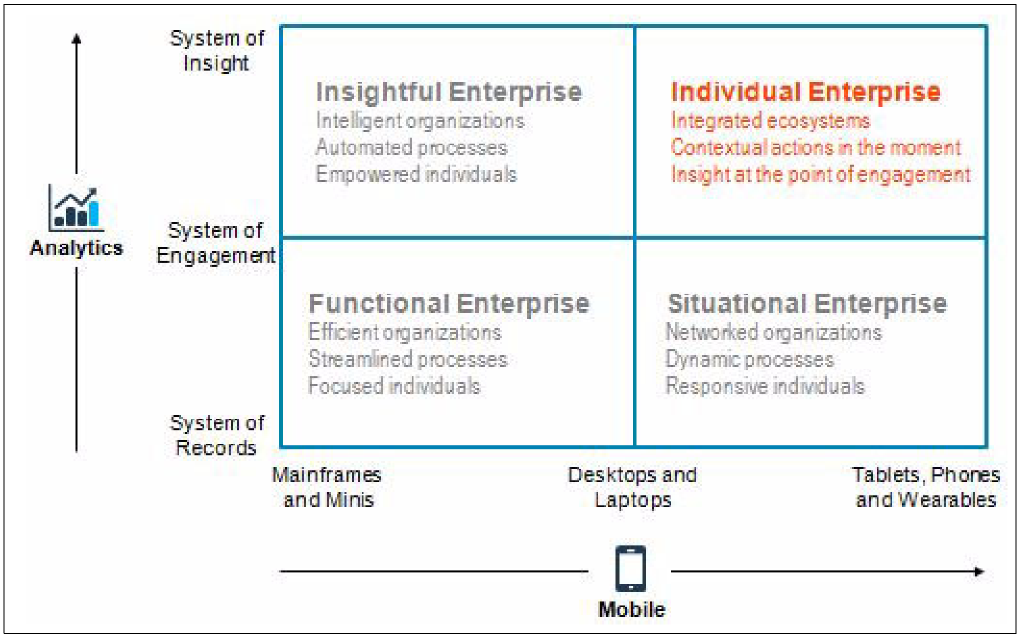


Figure 1-1 Analytics and mobile: The individual enterprise

1.2 IBM z Systems platform: A unique analytics implementation

As clients consider the many aspects of their analytics strategy and strive to keep pace with new and changing requirement, many are faced with growing IT complexity. Databases, core systems, the extraction, transformation, and loading (ETL) of data, replication, duplication,

[8] *Accenture Payment Services: Real-time payments for real-time banking,* October 2015

synchronization, departmental systems, localized analytics, security issues, and incremental growth are all factors that add to the growing complexity. This is not complexity by design, this is only the evolution of a computing infrastructure that was never designed to support what the organization now considers business-critical analytics.

1.2.1 Islands of analytics

Many customers have tried to move their data to their analytics solutions, putting their analytics in a separate environment from their transactional systems, which immediately introduces latency and islands of analytics. When the data is taken off the system of record where the transactional data originates, different groups will see snapshots of data that are not fresh and there is no guarantee of consistency in data synchronization. Multiple copies of the data must also be supported, which can lead to data security concerns.

Adding together all of these issues, excessive costs (including personnel costs) and inefficiencies will be the end result. This is a typical challenge that most clients are dealing with today as they re-evaluate their analytics strategy to support analytics that are now highly critical to the success of the business.

1.2.2 A hybrid transaction and analytics platform approach

The IBM z Systems platform provides a truly modern, cost-competitive analytics alternative that is primed to embrace the market shift around big data, mobile, customer interaction, and cloud initiatives. With the z Systems analytics, organizations can apply the same qualities of service to their business-critical analytics as they do to their transactional systems. An estimated 80% of corporate data is stored or originates on the platform.[9] Instead of moving the data to the analytics, the z Systems platform is a hybrid transactional and analytics environment, enabling analytics to be co-located with the data thus alleviating many of the complexities highlighted previously. Organizations can start with their most pressing analytics issues, quickly realize immediate business value, and then position their analytics strategy to grow and evolve along with business and market demands, all without the need to re-architect.

With the IBM z Systems platform, organizations can perform prescriptive, predictive, investigative, cognitive, capacity management analytics; this can be done all on one platform while being able to integrate with many other remote sources of data and applications on other platforms. IBM has ported its Cognos®, SPSS®, BigInsights® solutions to the platform with Apache Spark, which is the latest analytics platform available on Linux for z Systems and IBM z/OS operating systems.

1.2.3 Sparking a new chapter in analytics

While Apache Hadoop and Map Reduce provide a parallel computing cluster for the analysis of large and complex data sets, ease of use and timely execution of queries have sometimes been barriers to adoption, falling short in terms of user expectations and experience. In some cases, Hadoop implementations remain as a data lake or reservoir to store data for future analysis. Chapter 2, “Apache Spark overview” on page 5 describes Spark and how it aims to address complexity, speed, ease of use, and why it is key to the future success of analytics and the impact it is having and will continue to have on organizations, the business, their analytics strategies, and data scientists and developers.

[9] Mainframe Insights, March 2014 http://mainframeinsights.com/80-of-the-worlds-corporate-data-resides-or/

2

Apache Spark overview

Analytics is increasingly an integral part of day-to-day operations at today's leading businesses, and transformation is also occurring through huge growth in mobile and digital channels. Previously acceptable response times and delays for analytic insight are no longer viable, with more push toward real-time and in-transaction analytics. In addition, data science skills are increasingly in demand. As a result, enterprise organizations are attempting to leverage analytics in new ways and transition existing analytic capability to respond with more flexibility, while making the most efficient use of highly valuable data science skills.

Although the demand for more agile analytics across the enterprise is increasing, many of today’s solutions are aligned to specific platforms, tied to inflexible programming models, require vast data movements into data lakes. These lakes quickly become stale and unmanageable, resulting in pockets of analytics and insight that require ongoing manual intervention to integrate into coherent analytics solutions.

With all these impending forces converging, organizations are well-poised for a change. The recent growth and adoption of Apache Spark as an analytics framework and platform is timely and helps meet these challenging demands.

This chapter addresses the following topics:

- What is Apache Spark
- IBM strategy and added value to Apache Spark
- Apache Spark on z Systems platforms

2.1 What is Apache Spark

Apache Spark[1] is an open source, in-memory analytics computing framework offered by the Apache Foundation.

Spark offers a unified programming environment and is extremely lightweight. What is most important is that Spark is function-rich in that it provides libraries for commonly used analytic methodologies for data access, manipulation and application of various algorithms. Apache Spark offers language diversity in its support for Java, Python, Scala, and most recently, R.

From an operations perspective, Spark can run in stand-alone mode or clustered environments, and it is not reliant on a specific file system or platform set of technologies, but it can be adapted to many configurations.

One of the key aspects of Spark that has attracted a growing following of adopters and contributors is its strength as a unification of the programming interfaces for analytics. Spark is not only about data access, it is about the framework that is offered in terms of analytic programming context. In many ways, the analogy can be made between what Spark offers for analytics and how IBM WebSphere® and Java transformed application development. Consider the democratization of analytics interfaces, abstractions over underlying data access and implementation specifics, and also the variety of programming models to suit business needs.

Apache Spark is fundamentally structured in a way to provide independence of application environments (Figure 2-1).

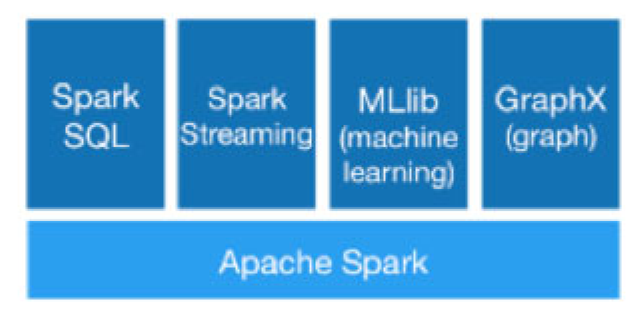


Figure 2-1 Apache Spark structure

The structure of Apache Spark has several key components:

- Spark Core

  The basis of the project provides task scheduling, dispatching, I/O, and the programming abstraction resilient distributed dataset (RDD), which is a read-only distributed collection of records that can be stored in memory or on disk. RDDs then offer various operations that can be performed on them including transformations such as map, union, intersect, and actions such as collect, count, or aggregate with a function.

- Spark SQL

  This function on top of Spark Core provides a further abstraction of RDDs used for accessing structured and semi-structured data, with access through SQL interfaces and access to sources of data via JDBC and ODBC interfaces. The data abstractions for RDDs used in this context are referred to as *DataFrames* (formerly known as SchemaRDDs).

[1] http://spark.apache.org

- Spark Streaming

  This component is suitable for ingesting data and leveraging the operations available for RDDs on ingested batches of data.

- Spark MLib

  This machine learning framework is on top of Spark Core and implements a number of commonly used machine learning algorithms (for example correlations, random data generation, classifications, regressions, decision trees, clustering, and others).

- GraphX

  This is a graph processing framework and an API, enabling efficient graph computations with interactions across both graph representations and RDD representations of the same data without duplication.

With all the challenges on enterprises and the advantages of Spark, it is understandable why the Spark community of both users and contributors has grown so significantly; even commercial vendors are now offering Apache Spark-based solutions for key analytics insight such as fraud detection and customer insight.

2.2 IBM strategy and added value to Apache Spark

In June 2015, IBM announced a major commitment to Apache Spark, including plans to put more than 3,500 IBM developers and researcher to work on Spark related projects worldwide, contribution of the IBM SystemML machine learning technology to the Spark open-source community, and intent to offer Spark as a service on IBM Bluemix®. IBM also established the Spark Technology Center with a focus on contributing features and function to the open source Spark community.

Over time, select technologies offered by IBM may use Apache Spark as part of the underlying platform to take advantage of Spark's flexibility, in-memory analytics and built-in set of machine learning libraries.

2.3 Apache Spark on z Systems platforms

You might hear much about the volume, variety, veracity, and volatility of data coming from sources that are external to the enterprise (such as social media, blogs, Twitter, and others). However, insight that can be gained by combining the analytic results from external data with high value (and highly sensitive) data held within the enterprise can deliver superior results to the business. Often, this high-value data for enterprise customers resides on the z Systems platform.

In the past, leading practices for gaining insight from multiple sets of data necessitated moving all this data into one location for purposes of simplifying the analytic programming environment and correlating across multiple data environments. This data centralization strategy resulted in negative business consequences of these vast data transfers: data latency, analytic latency, data security, governance, cost, availability, auditability, and of course risk of exposure to breach. However, leading practices can and should change when a better solution presents itself.

Spark offers many advantages in its federated analytics approach; however, think about the greater potential advantages to RDDs that reside in memory governed by a secure z Systems environment, performance that is optimized not for only data access but also execution of the analytics required on that data. Conversely, consider the potential risk of accessing z Systems high-value business data remotely and having this data available across many distributed systems through RDD memory structures. With the unified analytic framework described, enabling Spark natively on z Systems platforms can allow these emerging analytic applications to use data-in-place: in an environment that offers locality of reference, secure governance and optimized implementation.

These key values to enterprises is why IBM Systems has enabled Spark natively for both z/OS and Linux on z Systems. Apache Spark is enabled on both of the operating system environments supported on z Systems hardware; clients can choose the configuration that fits best with their needs. The suggestion is to consider the originating sources of data and transactions that will feed the Spark analytics. If most of the data that will be used for Spark analytics, or the most sensitive or quickly changing data is originating on z/OS, then a Spark z/OS based environment will be the optimal choice for performance, security, and governance. If most of the data that will be used for Spark analytics originates on Linux on z, then a Linux on z Spark is a viable approach.

Of course, not all the data needs to be hosted in one platform. In fact, the strength of Spark is that it can combine data from a wide variety of heterogeneous data sources and provide a clean data abstraction layer.

Consider one use case (Figure 2-2 on page 9) that uses Spark-based analytics for determining whether the consumer that is associated with a credit card transaction is a good candidate for a promotional offer, combining insight from sensitive transaction and account information analytics with insight from sentiment analysis based on the consumer's social media content.

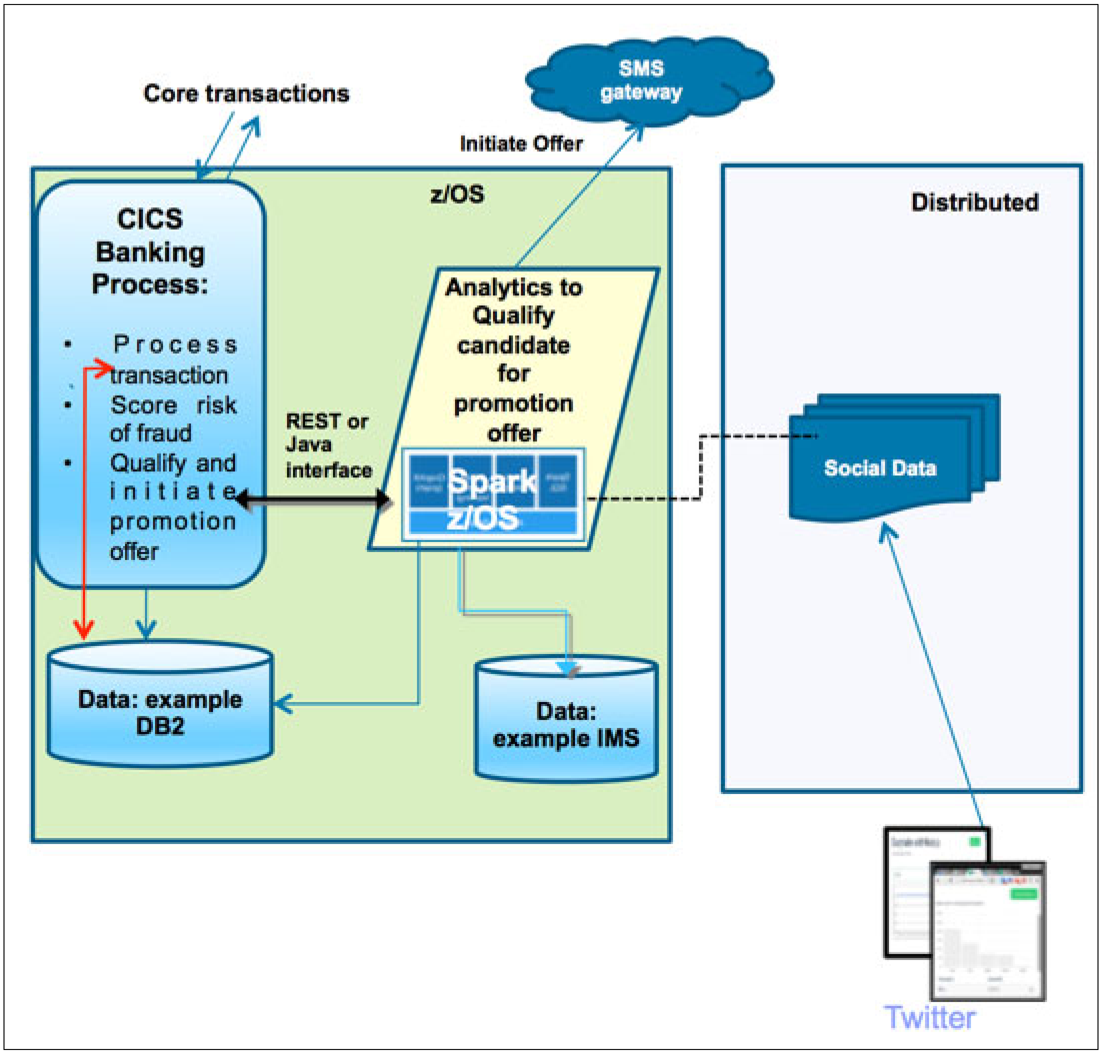


Figure 2-2 Integration of Apache Spark z/OS with transaction systems

With this method, z/OS transaction and business data is analyzed in place securely, and the relevant information is associated with insight from unstructured analytics on external data such as that from social media. The insight is exchanged without the movement of data. With Spark, you can federate the insight, not centralize the data, to achieve superior business results.

Through leveraging Spark's consistent interfaces and rich analytics libraries for creation of analytic content, data scientists and programmers can quickly build high-value analytic applications across multiple environments (including z Systems platforms) that use data in place and federate the analytic processing to best fit environments. Spark analytics can offer both batch and real-time capabilities, depending on the desired qualities associated with the analytics.

Next, examine more closely the structure of an Apache Spark environment on z/OS. Figure 2-3 shows a high level structure of the integration of Apache Spark on z/OS.

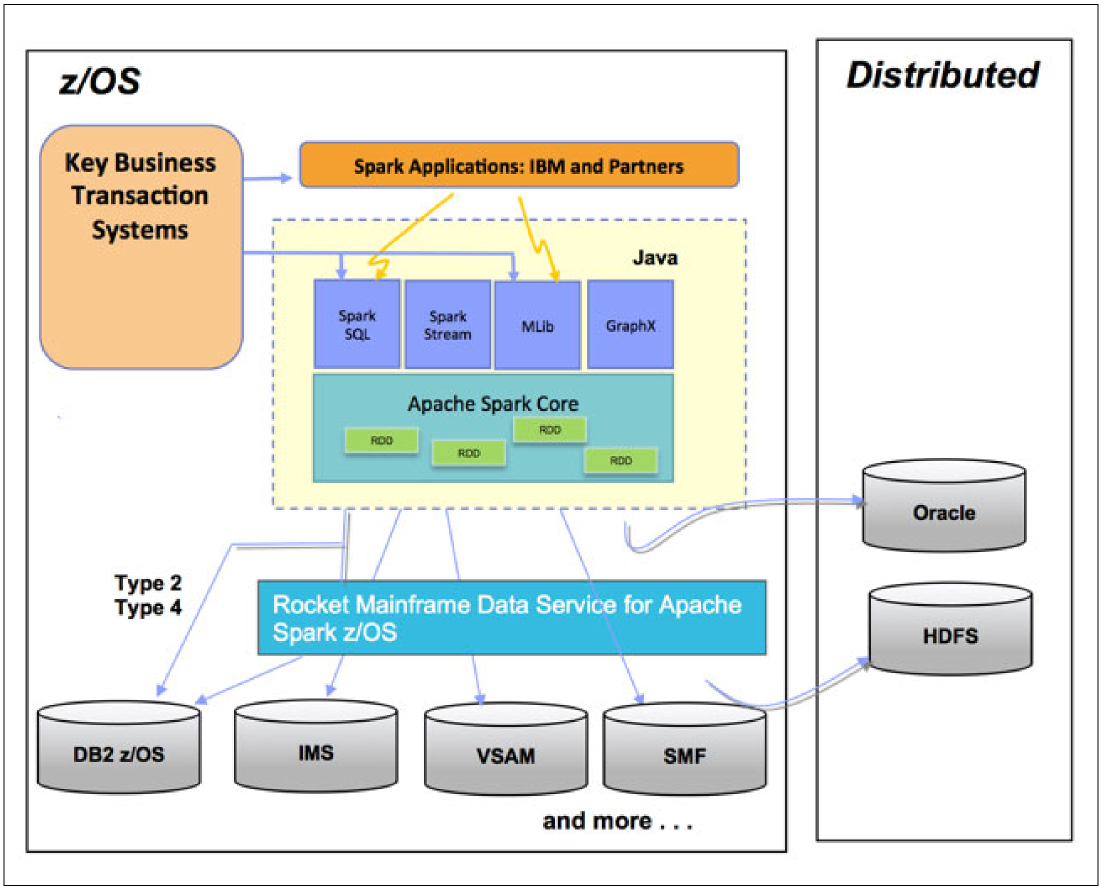


Figure 2-3 Integration of Apache Spark z/OS

This structure shows one Spark environment running natively on z/OS. Spark can also be clustered across more than one JVM, and these Spark environments can be dispersed across an IBM Parallel Sysplex®.

Because Spark is based on Java, the potential exists for z Systems transactional environments, customer-provided applications, and IBM and other vendor applications to leverage the consistent Spark interfaces with almost all zIIP-eligible MIPS. In this way, analytics processing on z/OS becomes extremely affordable. With the IBM z13™ system, IBM supports up to 10 TB of memory that can enable the in-memory RDD Spark structures for optimal performance. Through the Spark SQL interfaces, access to DB2 z/OS and IMS can be facilitated through standard types 2 and 4 connections. Depending on the level of data integration that you want, access to VSAM, physical sequential, SMF, SYSLOG, and other environments is also possible.

Specific to z/OS Apache Spark, Rocket Software created a function named *Rocket Software Mainframe Data Service for Apache Spark z/OS*. This capability can enable Apache Spark z/OS to have optimized, virtualized, and parallelized access to a wide variety of z Systems and non z Systems data sources.

Spark on z Systems platforms is now available at the following location:

https://www.ibm.com/developerworks/java/jdk/spark/

3

Apache Spark at the core of the analytics platform

Apache Spark represents a unique opportunity to enrich the IBM Analytics Platform ecosystem. This chapter describes how Apache Spark constitutes an integral component of the IBM Open Platform with Apache Hadoop and IBM InfoSphere® BigInsights for Apache Hadoop for Linux on z.

This chapter addresses the following topics:

- The ODPi industry effort
- IBM Open Platform with Apache Hadoop
- IBM InfoSphere BigInsights for Apache Hadoop
- Information integration on Apache Hadoop
- Using Apache Spark for the IBM Analytics Platform
- Apache Spark and the broader analytics ecosystem

3.1 The ODPi industry effort

In February 2015, the Open Data Platform initiative (ODPi)[1] was announced by IBM, Hortonworks, Pivotal, EMC, SAS, Teradata, and VMware. Today, the ODPi has approximately 25 members. The ODPi is a shared industry effort focused on advocating and advancing the state of Apache Hadoop and a core Hadoop stack on which all suppliers can agree. One of the key objectives is to foster big data solutions by providing a well-defined core set of open source platform components. Furthermore, another key objective is to facilitate and certify a standard ODPi core of compatible versions of chosen big data open source projects. For additional goals and objectives, see the ODPi website.

The ODPi serves as the foundation for IBM to deliver its IBM Open Platform with Apache Hadoop.

3.2 IBM Open Platform with Apache Hadoop

IBM BigInsights delivers a rich set of advanced analytics capabilities that allows enterprises to analyze massive volumes of structured and unstructured data in its native format. The software combines open source Apache Hadoop with IBM innovations including sophisticated text analytics, IBM BigSheets for data exploration, IBM Big SQL for SQL access to data in Hadoop, and a range of performance, security, and administrative features. The result is a cost-effective and user-friendly solution for complex, big data analytics.

IBM BigInsights is available in two product editions:

- *IBM BigInsights for Apache Hadoop*: An industry standard Hadoop offering that combines the best of open source software with enterprise-grade capabilities.
- *IBM Open Platform with Apache Hadoop*: Builds the platform for big data projects and provides the most current Apache Hadoop open source content.

IBM Hadoop distribution includes the no-cost IBM Open Platform with Apache Hadoop, which is an industry-compatible Apache Hadoop distribution built to the ODPi specifications. It is available at no cost or with a paid support option. As Figure 3-1 shows, IBM Open Platform includes the Spark in-memory distributed compute engine with the Spark Core, Spark SQL, Spark Streaming, Spark MLlib, and GraphX components.

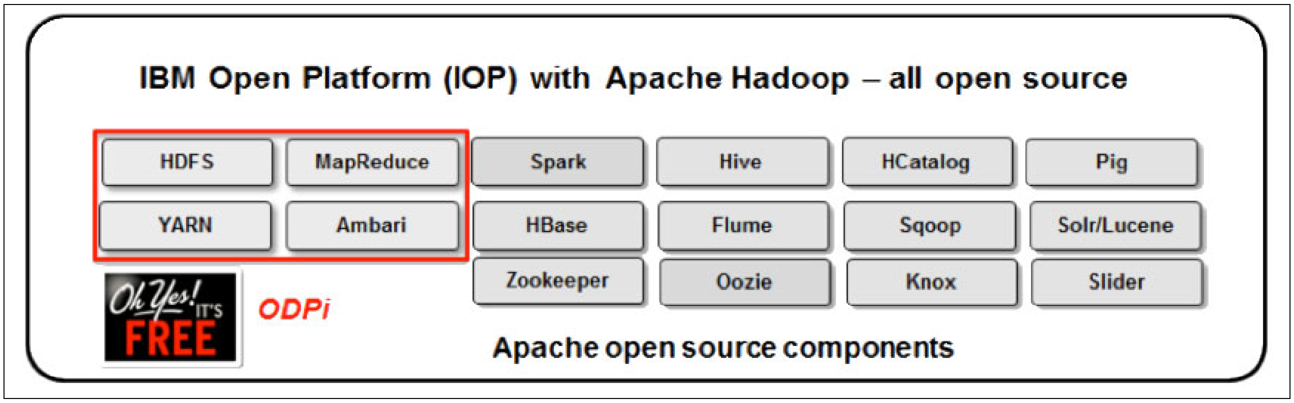


Figure 3-1 IBM Open Platform with Apache Hadoop

[1] ODPi: https://www.odpi.org/, retrieved January 2016.

In addition to Spark, the following Apache Hadoop projects are included in IBM Open Platform: Ambari, Apache Kafka, Flume, Ganglia, Hadoop (HDFS, YARN, MapReduce), HBase, Hive, Knox, Lucene, Nagios, Oozie, Parquet, Parquet Format, Pig, Slider, Snappy, Solr, Sqoop, Teradata Connector for Hadoop, and Zookeeper.

The IBM Open Platform fits seamlessly into an existing IT infrastructure, from IBM InfoSphere Information Server to IBM InfoSphere Guardium®. Integrate easily with other IBM offerings to quickly move data in and out of IBM InfoSphere, IBM Streams, and more.

3.3 IBM InfoSphere BigInsights for Apache Hadoop

IBM Open Platform with Apache Hadoop (Figure 3-2), the core of IBM BigInsights[2], is among the first Hadoop distributions inspired by the ODPi. The IBM BigInsights Analyst, Data Scientist, and Enterprise Management modules are value-added technologies built to run on the IBM Open Platform with Apache Hadoop, and are therefore designed to benefit from enhanced interoperability with other big data tools.

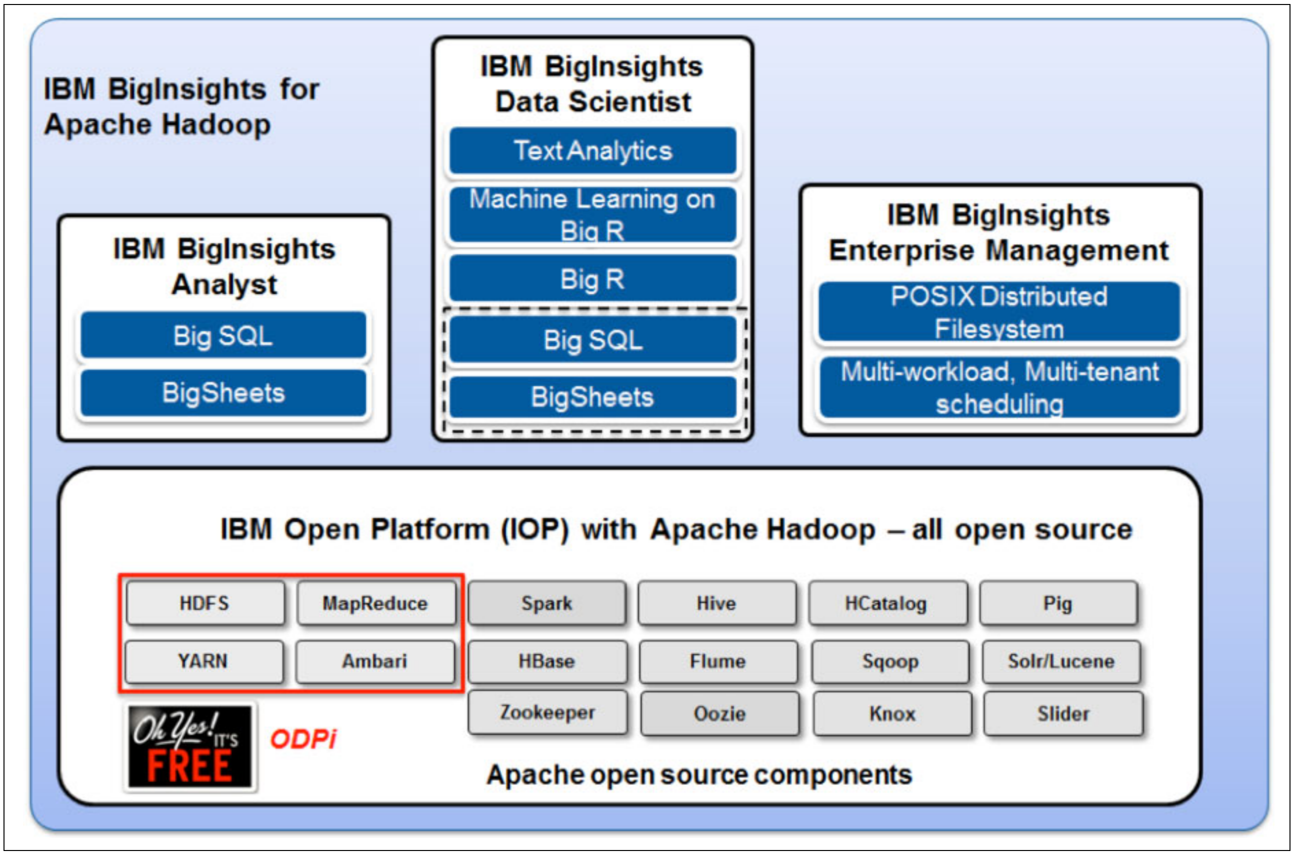

IBM BigInsights for Apache Hadoop
IBM BigInsights Analyst
Big SQL
BigSheets
IBM BigInsights Data Scientist
Text Analytics
Machine Learning on Big R
Big R
Big SQL
BigSheets
IBM BigInsights Enterprise Management
POSIX Distributed Filesystem
Multi-workload, Multi-tenant scheduling
IBM Open Platform (IOP) with Apache Hadoop – all open source
HDFS
MapReduce
YARN
Ambari
Spark
Hive
HCatalog
Pig
HBase
Flume
Sqoop
Solr/Lucene
Zookeeper
Oozie
Knox
Slider
Oh Yes! IT'S FREE
ODPi
Apache open source components

Figure 3-2 IBM InfoSphere BigInsights for Apache Hadoop

The newest Spark version as part of InfoSphere BigInsights for Apache Hadoop includes SparkR, which is an R binding for Spark based on Spark's new DataFrame API. SparkR gives R users access to Spark's scale-out parallel runtime along with all of Spark's input and output formats. It also supports calling directly into Spark SQL.

[2] IBM BigInsights, http://www.ibm.com/software/products/en/ibm-biginsights, retrieved January 2016

The following list briefly describes new capabilities in IBM BigInsights for Apache Hadoop:[3]

- Access to all data, whether in Hive, HBase, or HDFS within a single query (Big SQL)
- Improved high availability, greater performance, and richer SQL (Big SQL)
- Ability to manipulate and visualize data with a spreadsheet-like interface that now includes web tooling for business users (BigSheets)
- Automated prediction through machine learning algorithms in R
- Deeper insight through advanced analytics, including text and geospatial
- Enhanced text analytics that can infer context and relationships from text

Several products are related to BigInsights:

- IBM BigInsights on Cloud
- IBM BigInsights BigIntegrate
- IBM BigInsights BigQuality
- IBM InfoSphere Big Match for Hadoop

BigInsights BigIntegrate and BigInsights BigQuality especially are essential tools to perform information integration and preparation tasks on z Systems platforms prior to conducting analytical tasks with Spark on the mainframe.

3.4 Information integration on Apache Hadoop

Although Apache Spark on z/OS and Spark for Linux on z can use data in place and federate the analytical processing, in several use case scenarios, data from various z Systems subsystems (for example, VSAM, IMS) and non z Systems sources (for example Twitter, DB2 LUW (Linux, UNIX, Windows), Oracle) must be adequately prepared, cleansed, and transformed, redistributed, and restructured according to downstream consuming system requirements prior to performing Spark analytics. This information integration aspect will most likely persist, although federation and real-time analytics will complement the analytics patterns as well.

In those cases, IBM offers data integration with BigInsights BigIntegrate and BigInsights BigQuality.[4] Built on the existing IBM InfoSphere Information Server, BigInsights BigIntegrate and BigInsights BigQuality are designed to bring required integration, quality, and governance capabilities to Apache Hadoop and Spark. BigInsights BigIntegrate and BigInsights BigQuality offer a complete set of data connectivity, transformation, cleansing, enhancement, and data delivery features that are immediately available to be deployed on various Hadoop distributions, including IBM Open Platform.

BigInsights BigIntegrate and BigInsights BigQuality provide the end-to-end information integration and governance capabilities that allow organizations to accomplish these goals:

- Understand data
- Cleanse, monitor, transform and deliver data
- Collaborate across the organization and bridge the gap between business and IT

Whether that information resides on-premises in a traditional relational DB or an existing Enterprise Data Warehouse (EDW), in a Hadoop-based data reservoir, or in the cloud, BigInsights BigIntegrate and BigInsights BigQuality enable line-of-business organizations to

[3] IBM Data Sheet: IBM BigInsights for Apache Hadoop – Accelerate analytics and data science with open source Hadoop, March 2015

[4] IBM Solution Brief, IBM BigInsights BigIntegrate and BigInsights BigQuality Information empowerment for your big data ecosystem, September 2015

implement a vast number of use-case scenarios addressing business requirements. These engine-exploiting, in-memory features, such as data partitioning, data pipelining, and dynamic repartitioning, seamlessly integrate with Hadoop YARN.

BigInsights BigIntegrate and BigInsights BigQuality support Apache Hadoop distributions that conform to the Open Data Platform (ODP) requirements such as IBM Open Platform, IBM BigInsights, and HortonWorks, and also to non-ODP distributions such as Cloudera.

z Systems centric analytics using Spark results in scenarios where the analytics and the data preparation tasks can be performed entirely on z Systems platforms. Thus, regardless of the characteristics and requirements of a specific analytics use-case scenario, in terms of source data structures, analytical insight required, heterogeneous source systems landscape, and information integration needs, the z Systems platform is a pervasive platform for a vast number of use cases.

3.5 Using Apache Spark for the IBM Analytics Platform

Apache Hadoop and Apache Spark constitute a significant value for some IBM analytics products by leveraging and integrating open source components. This Apache Hadoop and Spark story goes far beyond just ODPi, IBM Open Platform, and BigInsights. Across the IBM Analytics Platform product and tools portfolio, almost every major software offering has integrations with Hadoop and Spark. For example, IBM Watson™ Analytics[5] uses Hadoop for large scale storage and data processing; InfoSphere Big Match for Hadoop[6] enables large scale matching to be done on Hadoop instead of relational databases, and also enables entity extraction from text sources and the ability to match those entities with relational data.
The IBM Information Server ETL engine will run as a YARN application on many Hadoop distributions, including IBM Open Platform, meaning that it will be a “first-class citizen,” deeply integrated in Hadoop.

Also of note here is IBM InfoSphere Streams[7], which is the industry’s leading real-time streaming data processing engine. It can write to Hadoop file systems, and even spawn Hadoop applications for at-rest processing.

3.6 Apache Spark and the broader analytics ecosystem

The developer story across this IBM Analytics Platform stack is also notable because common code can be run in multiple settings. As examples, SPSS models can be run on Streams, Hadoop, or IBM PureData® for Analytics; AQL text extractors can be run on InfoSphere Streams, or Apache Hadoop; R applications can be run on Streams, Hadoop, or PureData for Analytics.

The big story here is that to enable a vision for big data, the entire IBM stack for analytics integrates and supports Hadoop and Spark. This way enables users to benefit from the scalability and flexibility of technologies like Hadoop and Spark, while still enjoying their familiar interfaces (like SQL, Cognos, or R) and advanced analytic tools (like Watson Analytics). These business users and data scientists need worry only about their analytics and data exploration activities, without having to learn new and evolving technologies.

[5] IBM Watson Analytics, https://www.ibm.com/marketplace/cloud/watson-analytics/us/en-us, retrieved January 2016.

[6] IBM, Solution Brief, IBM InfoSphere Big Match for Hadoop – Accurately connect structured and unstructured customer data for deep insights and effective decision-making, October 2015.

[7] IBM InfoSphere Streams, http://www.ibm.com/software/products/en/ibm-streams, retrieved January 2016.

Providing analytics in the z Systems context requires the support of different analytics use-case patterns. Some of these patterns require data movement and integration; others might rely on keeping data in place. Whatever the underlying characteristics are, Spark enriches z Systems analytics scope. For instance, the Apache Spark integration in SPSS Modeler includes optimization of SPSS algorithms for Apache Spark, and extension of the SPSS Modeler with Spark algorithms. Apache Spark furthermore facilitates required integration points. For instance, having Apache Spark as the analytics computing framework of choice will significantly expand the entire z Systems analytics ecosystem, and will provide transparency for development and deployment of z Systems centric analytics use case scenarios.

4

Federated analytics: Apache Spark integration with other technologies

One of the main advantages of Apache Spark lies in its ability to perform federated analytics over a heterogeneous source data landscape. This chapter explores how Apache Spark can be integrated with the following enterprise technologies:

- DB2 for z/OS
- DB2 Analytics Accelerator
- IMS Database Manager
- CICS Transaction Server
- Other data sources

4.1 DB2 for z/OS

Apache Spark promises to be a “game-changer” for big data by providing a unified analytics platform and is emerging as a *de facto* “analytics operating system.” The Spark ecosystem is continuously growing with different products in the big data space leveraging Spark as their underlying execution engine. Spark has quickly evolved into the hottest open source project because of its potential to solve complex big data problems in a very simple way; it cuts through the complexity of MapReduce and provides developer-friendly Scala, Java, Python and R APIs suited for both interactive and batch processing. It also provides rich libraries for machine learning, streaming, graph processing, and statistical analysis.

Because it is able to support a wide variety of structured and unstructured data sources, Spark is positioned to be the enterprise-wide analytics engine. Most enterprises store data in heterogeneous environments with a mix of data sources. With Spark, the tasks have become easier than ever to ingest data from disparate data sources and perform fast in memory analytics on data combined from multiple sources, giving a 360-degree view of enterprise-wide data.

Big data is not all about unstructured data. In fact, most real-world problems are solved using some form of structured or semi-structured data. Because DB2 for z/OS is the market leader for enterprise-structured data, an integration of Spark with DB2 is an obvious next step in the evolution of big data. Enterprises store petabytes of transactional data in DB2 and run their mission-critical applications on that data. Customers often have a need to perform analytics on not just pure DB2 data, but aggregate DB2 data with other data sources to derive additional business insights. For example, a business might want to aggregate transactional data in DB2 with social media data, such as Twitter data stored in HDFS, to establish patterns on consumer sentiment and take actions such as offering targeted discounts. Combining Spark and DB2 simplifies integration of mission critical transaction data with contextual data from other sources to derive additional Big Data insights.

Spark provides an easy integration with structured data using SparkSQL, a module in Apache Spark that integrates relational processing with Spark’s functional programming API. Spark SQL Data Sources support is helpful to more simply connect to relational databases, load data into Spark, and also access Spark data as though it were stored in a relational database. SparkSQL lets Spark programmers leverage the benefits of relational processing and lets SQL users call complex analytics libraries in Spark, such as machine learning.

Spark, which is built with Scala, enables large scale data processing by abstracting data as collection of objects, referred to as *resilient distributed datasets* (RDDs), spread across clusters. An extension to RDD as part of SparkSQL, known as DataFrames, enriches RDDs with schema, so that data engineers and scientists can more easily work with large data sets. DataFrames can be considered in memory relational tables, so that people with an SQL background can more easily perform analyses with Spark.

SparkSQL is viewed as the unified way to access structured data in the Spark ecosystem. DataFrames support a wide variety of data formats that are ready to use, such as JSON and Hive. It can also read and write to external relational data sources through a JDBC interface. This ability of DataFrames to support a wide variety of data sources and formats enables rich federation of data across many sources. DB2 offers integration with Spark SQL using the DB2 JDBC driver. DataFrames API allows loading of DB2 data into Spark through the JDBC driver so that exposing DB2 data as Spark DataFrames happens more easily. SQL queries can be run on a DataFrame instantiated with DB2 data. DataFrames also provides abstraction for selecting columns, joining different data sources, aggregation, and filtering. After DB2 data is loaded into Spark as DataFrames, those DataFrames can be joined with data from other sources, or transformations can be applied to generate new DataFrames. Transformed

DataFrames can even be written back into DB2 and persisted. All this can be done through SQL, or rich language bindings in Python, Scala, Java and R. Data scientists can go beyond joins, aggregation, and filtering on DataFrames created from DB2 data: they can even use complex user functions on DataFrames for advanced analytics and also MLib's machine learning pipeline API for machine learning. See Figure 4-1.

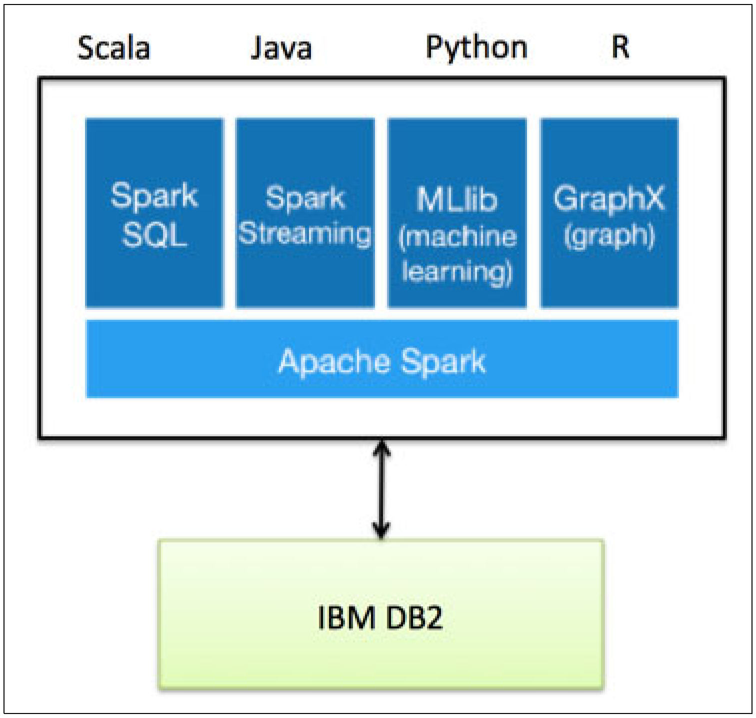


Figure 4-1 Integration of DB2 data with Apache Spark

The seamless integration of DB2 data with Spark offers business benefits to customers by allowing them to easily enrich mission-critical DB2 data with external data and take business decisions on the holistic view of data. DB2 data can be analyzed along with data stored in other data sources, such as Amazon Redshift, HDFS, MySQL, Oracle, Postgres, SQL Server, and others. Spark allows a single program to perform ETL and analytics on DB2 data combined with other sources of data. It also provides an easy solution to integrate machine learning and streaming with DB2 data sets, which might be difficult to do with pure RDMS and SQL. IBM is committed to taking DB2 and Spark integration to the next level by introducing optimizations and richer API integrated in Spark releases. Spark is powering a big data revolution and DB2 will be a first-class citizen in the Spark ecosystem. Their integration can enable customers to leverage best-in-class advanced analytics on mission-critical enterprise data in DB2.

4.2 DB2 Analytics Accelerator

The IBM DB2 Analytics Accelerator for z/OS is an innovation that is leading the way to support highly complex analytical workloads using data that resides on IBM z Systems platform. The Accelerator provides the foundation for analytics on z Systems environments. This section provides a high-level overview about the DB2 Analytics Accelerator and illustrates the role of the Accelerator in a Spark federated analytics landscape.

4.2.1 Overview of the DB2 Analytics Accelerator

The DB2 Analytics Accelerator[1] is a high-performance accelerator for z Systems platforms that supports data-intensive and complex queries. Complex multidimensional queries can run as much as 2,000 times faster than the same query running natively on IBM DB2 for z/OS. One DB2 for z/OS handles transactional queries while the Accelerator performs analytical queries. The Accelerator is hidden to external callers, so users and applications see only a DB2 for z/OS interface.

Many analytics usage patterns[2] can be implemented by using the DB2 for z/OS with the Accelerator and a pervasive value proposition:

- Enabling analytical queries on operational data, leveraging the radical performance improvements
- Building an EDW on the z Systems platform and accelerating complex, multidimensional DW queries
- Reducing the number of data marts and making data marts obsolete over time, thus reducing the number of multiple copies of data
- Implementing operational data stores (ODS) for operational reporting and analytics on transaction data
- Enabling data reservoirs[3]
- Performing analytics and SQL on IBM IMS, VSAM, and sequential data
- Leveraging embedded analytics by using in-DB predictive analytics (SPSS integration) in the Accelerator
- Simplifying the information supply chain by using the in-DB transformation in the Accelerator and reducing the complexity of the enterprise-wide data flow
- Analyzing SMF Logs by using the Accelerator with IBM Tivoli® Decision Support for z/OS and the IBM DB2 Analytics Accelerator Loader for loading SMF Log records directly into the Accelerator
- Implementing a data scientist work area and being able to create temporary DB objects for ad hoc analysis, leveraging accelerator-only tables (AOTs)
- Improving performance for iterative campaign tuning for IBM Campaign (formerly Unica® Campaign)
- Improving performance for iterative reporting and enabling deeper insight into operational status through faster reporting, for instance for MicroStrategy, IBM QMF™, homegrown applications, and so on
- Offloading historical data from DB2 for z/OS on DB2 Analytics Accelerator

4.2.2 DB2 Analytics Accelerator in the context of Apache Spark

The Accelerator already enables many z Systems centric analytics use cases. In the context of Apache Spark, the Accelerator underpins the Spark federated analytics paradigm by optimizing the access to data through DB2 for z/OS.

In an Apache Spark environment on z/OS, a substantial amount of relevant data resides in DB2 for z/OS. Access to this data through Spark SQL is provided by using the Spark

[1] IBM DB2 Analytics Accelerator for z/OS, http://www.ibm.com/software/products/en/db2analacceforzos, retrieved January 2016.

[2] *Accelerating Data Transformation with IBM DB2 Analytics Accelerator for z/OS*, SG24-8314.

[3] M. Chessell, E. Hechler, *IBM DB2 Analytics Accelerator and the data reservoir*, IBM White Paper, November 2015.

DataFrames API. Thus, data in DB2 for z/OS can be exposed as RDDs and processed for instance via Spark SQL or Spark MLlib algorithms. The Accelerator underpins this data access method, allowing fast retrieval of data through DB2 for z/OS. In addition, data from other z/OS subsystems, such as VSAM records, or data from other RDBMS systems that has been moved into the Accelerator for fast SQL retrieval through DB2 for z/OS can be exposed as Spark RDDs and further processed by any of the Apache Spark components, for instance Spark MLlib.

Especially for large data volumes, where the Accelerator was used to increase the speed of data retrieval via DB2 for z/OS, this infrastructure can be used to deliver superior performance with Apache Spark analytics on relational data. As outlined in Chapter 2, "Apache Spark overview" on page 5, Apache Spark provides federated analytics through the Rocket Software Mainframe Data Service for Apache Spark on z/OS.[4] This includes access to data from IMS, VSAM, and other z/OS subsystems, SMF Log records, and HDFS data.

The DB2 Analytics Accelerator underpins and complements these scenarios where some of the data, for instance large volume of VSAM and IMS data, or SMF Log records do already reside in the Accelerator, and where the Accelerator provides required performance in retrieving subsets and even pre-aggregated data into Apache Spark analytics landscape.

4.2.3 Trends and directions

IBM z Systems and DB2 for z/OS platforms have traditionally been associated with OLTP and transactional workloads. The introduction of DB2 Analytics Accelerator has greatly increased the overall analytics on the z Systems value proposition; customers are increasingly running analytical workloads on z Systems platforms, and are deploying DWH systems using DB2 for z/OS with the Accelerator (Figure 4-2).

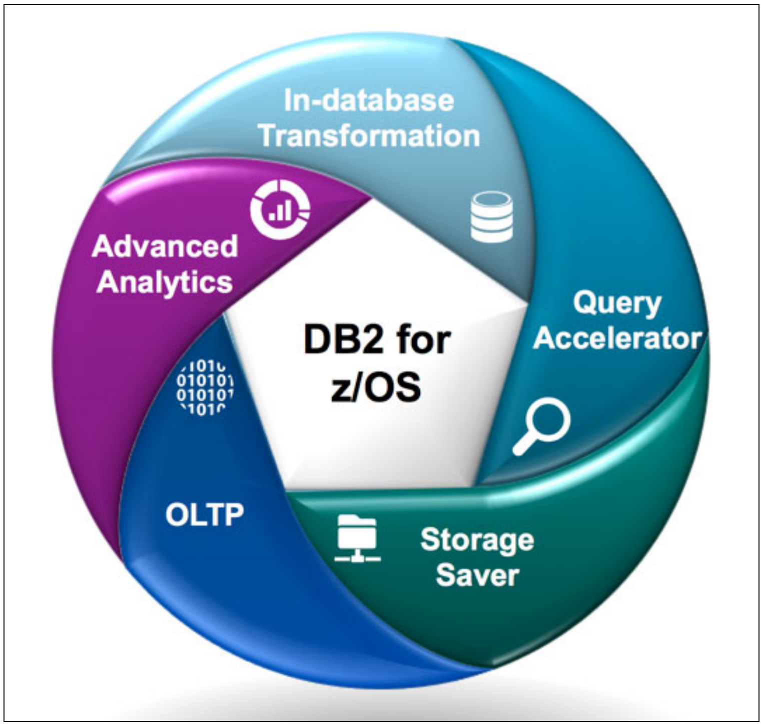


Figure 4-2 IBM DB2 Analytics Accelerator trends and directions

[4] Rocket Software, Rocket Mainframe Data Service for Apache Spark on z/OS, http://www.rocketsoftware.com/rocket-mainframe-data-service-apache-spark-zos, retrieved January 2016.

Moving forward, the Accelerator can further improve transparency to applications. It will focus on capabilities to ultimately allow the consolidation and unification of transactional and analytical data stores in a unified way. This will deliver the promise of the IBM Hybrid Transactional and Analytical processing paradigm.

In addition, the Accelerator reuses industry-leading IBM PureData® System for Analytics query and analytics capabilities and takes advantage of future enhancements, for instance in-DB analytics enhancements. Apache Spark might be a possibility to be leveraged as a capability within the PureData System for Analytics. This will enable even better performance to perform Apache Spark analytics closer to where the data resides.

Furthermore, the Accelerator can provide support of new use cases and roles, such as for data scientists, who are eager to use Spark Analytics themselves, with limited dependency from their IT organizations.

4.3 IMS Database Manager

The IMS Database Manager[5] is widely used by z Systems customers. IMS DB runs exclusively on the z/OS platform and uses hierarchical data structures. Clients continue to use IMS DB because hierarchical DBs can provide a significant performance edge over relational DB2 when queries are known in advance.

One of the advantages of Apache Spark lies in its ability to perform federated analytics over a heterogeneous source data landscape. In a z Systems environment, IMS data represents a significant business-relevant data source. Thus, performing Spark analytics on IMS data represents a unique opportunity to gain additional business insight.

With the Spark SQL interfaces, access to IMS data can be facilitated through a standard JDBC connection. As shown in Figure 4-3, the Apache Spark DataFrames API is used. This access is similar to the access to DB2 for z/OS data.

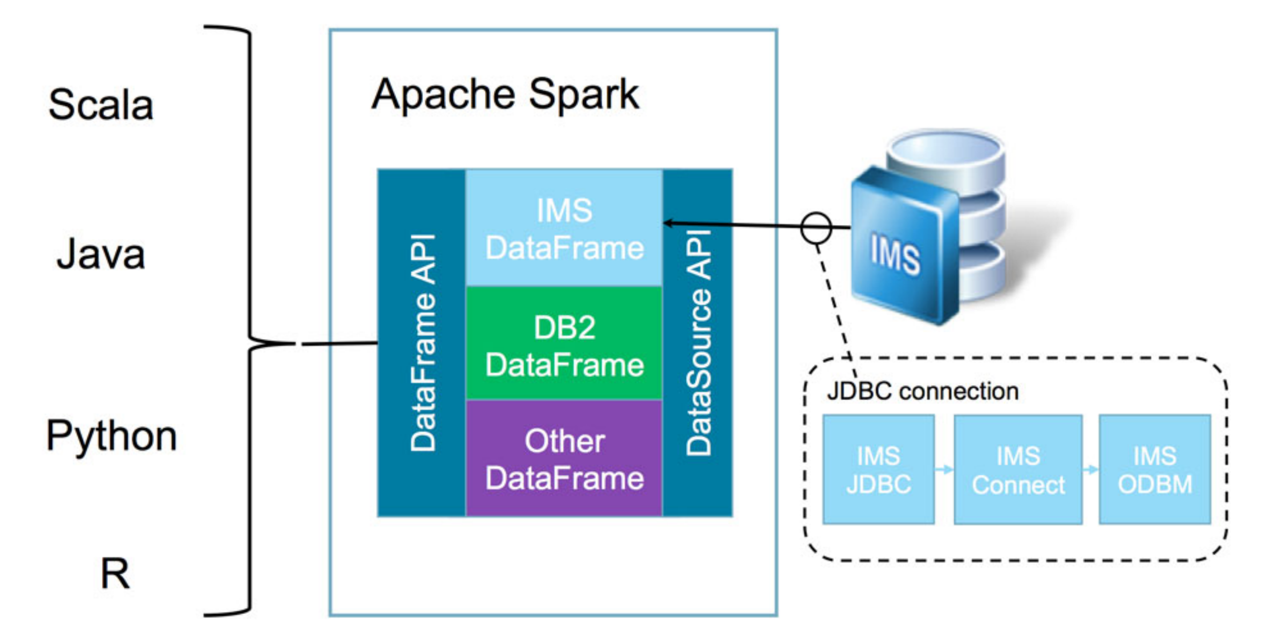


Figure 4-3 Importing IMS Data into Apache Spark

[5] IBM IMS, http://www.ibm.com/software/products/en/ims-product, retrieved January 2016.

IMS data can also be loaded into the DB2 Analytics Accelerator. Depending on the use case and analytical insight required, clients store IMS data into the Accelerator using a schema definition on DB2 for z/OS to access the IMS data with SQL through DB2 for z/OS. In those situations, where IMS data is already loaded into the Accelerator, the Spark SQL interface can be used to access IMS data through DB2 for z/OS.

4.4 CICS Transaction Server

CICS Transaction Server for z/OS (CICS TS)[6] is an enterprise-grade mixed-language application server that is used by organizations in almost all industries to process high volumes of business-critical transactions. These organizations have a growing need for their business transactions to be able to respond in real-time to the information and insight that can be provided by analytics.

Apache Spark offers the opportunity for transactions running in CICS TS to request scoring or other analytic results while processing requests, and to derive a different outcome for each request depending on the results returned by the analytics. The analytic processing can combine information that is available in the transactions, as they are running, together with other data that it can access from various sources.

Here are two examples:

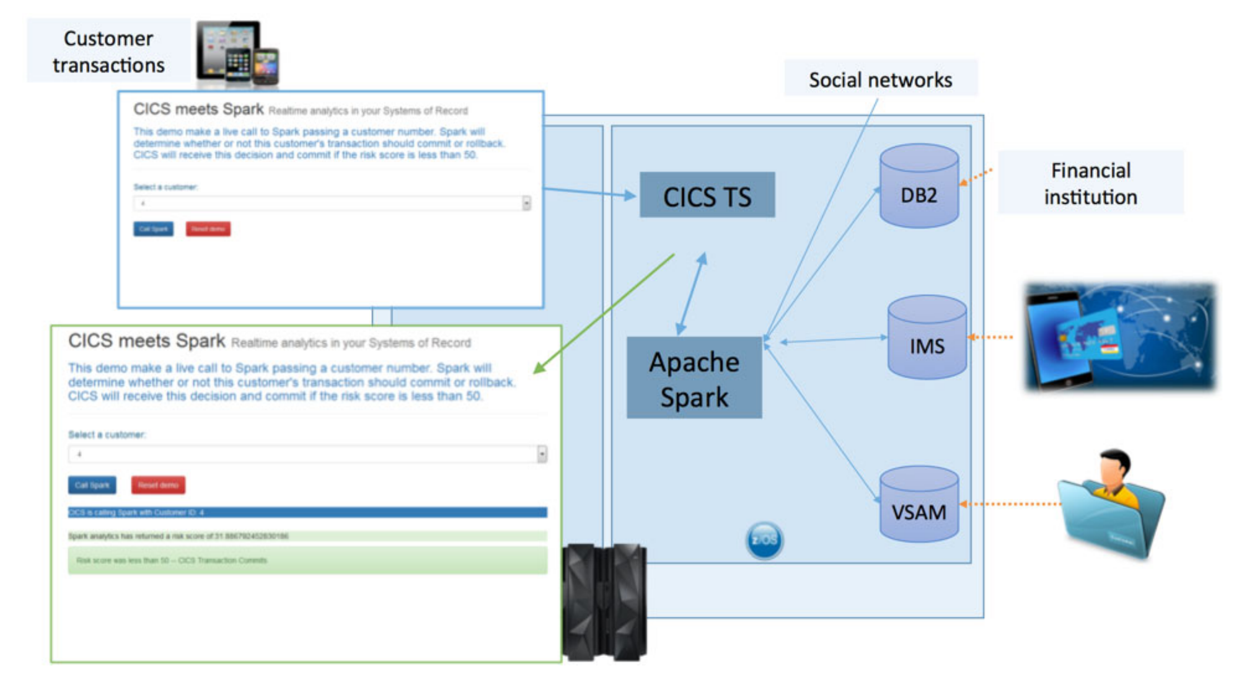
- A CICS application that is processing a retail purchase might invoke a Spark analytics model, passing it information about the customer and the transaction, which might be used in conjunction with historical information held in data stores and recent updates from social media, to decide whether the purchase is eligible for a discount. Applying the discount in this way, at the time of purchase as the transaction is processed, could give increased customer satisfaction over sending out speculative offers that customers might be able to use in the future.
- A credit card authorization that is processed by a CICS transaction might make a call to Spark to determine the likelihood that this is a fraudulent request. The transaction might then take appropriate action, such as rejecting the authorization by failing the transaction, or alternatively logging the need for further investigation.

In both examples, there might be latency benefits to running the Spark analytics on z/OS, co-located with where CICS is processing the requests.

[6] http://www.ibm.com/software/products/en/cics-tserver-zos

4.4.1 Invoking Spark Analytics from CICS TS

Figure 4-4 shows a simple example of CICS invoking Spark analytics from a transaction.



CICS meets Spark Realtime analytics in your Systems of Record

This demo make a live call to Spark passing a customer number. Spark will determine whether or not this customer's transaction should commit or rollback. CICS will receive this decision and commit if the risk score is less than 50.

Select a customer:

4

CICS meets Spark Realtime analytics in your Systems of Record

This demo make a live call to Spark passing a customer number. Spark will determine whether or not this customer's transaction should commit or rollback. CICS will receive this decision and commit if the risk score is less than 50.

Select a customer:

4

CICS is calling Spark with Customer ID: 4

Spark analytics has returned a risk score of 31.886792452830186

Risk score was less than 50 -- CICS Transaction Commits

Figure 4-4 Invoking Spark from CICS Transaction Server

In this example, details about the customer are passed to Spark, and the analytic processing uses data from various sources to assign a risk score to that customer's request. The risk score is used to determine whether the transaction should be allowed to complete, or whether it should be rolled back.

The call to Spark from CICS is made by issuing a REST request from a CICS Liberty Java program. The CICS program acts as a servlet, and issues a RESTful call to the Spark analytics application, which assesses the risk score. An HTTP connection is established, specifying the host, port, and URL of the Spark service, and an HTTP POST request is then issued to that connection. The customer identifier is included as a parameter on the request. The response from the Spark service can be obtained by retrieving an Input Stream from the java Connection object, and looping over this to get the contents of the HTTP body from the RESTful call to Spark. In the example shown, the risk returned by Spark is printed out by the servlet; in this particular case, a low-risk score resulted in the transaction being committed.

4.5 Other data sources

Accessing multitudes of heterogeneous data and performing analytics over this data using a consistent framework is one of the strengths of the Apache Spark platform. Clients using Apache Spark on z Systems platforms will want to analyze data from various sources, some non-relational and either unstructured or semi structured in addition to structured data. Examples of such data sources on z/OS include but are not limited to VSAM, PDSE, SYSLOG, and SMF. Examples of data sources in other environments that clients might want to include are Oracle, Teradata, and various Hadoop implementations.

In some situations, the use cases are based on business analytics, and in other cases, applications might exist that are more aligned with IT or systems analytics.

4.5.1 Implementation

Specific to Apache Spark running on z/OS, Rocket Software created the *Rocket Software Mainframe Data Service for Apache Spark z/OS* function. This capability enables Apache Spark z/OS to have optimized, virtualized, and parallelized access to a wide variety of z Systems and non z Systems data sources. For example, clients can configure the Mainframe Data Service for Apache Spark z/OS to access and map VSAM, SYSLOG, PDSE, and other data sources. After it is mapped, Apache Spark z/OS can be used through any of the Spark interfaces to read these data elements into RDD structures, and subsequently act on them with Spark analytics.

For example, a client can map VSAM data through the use of COBOL copybooks with the Mainframe Data Service for Apache Spark z/OS. After that is done, Apache Spark z/OS can access the VSAM data through SparkSQL, and perform analytic functions on that data that is found in Spark. The same process is used for any data source, and specific drivers are provided to be used by the Mainframe Data Service for Apache Spark z/OS for the various supported data sources. These drivers are then included in the Spark jobs that need access to the specific data sources.

Accessing non z/OS data sources is done through a similar implementation. Essentially the data source is configured and mapped in to the Mainframe Data Service for Apache Spark z/OS, a driver is provided and included in the Spark analytic job. Note that in this case, the z/OS system that is running Apache Spark and the Mainframe Data Service for Apache Spark z/OS must have network access to the distributed data source.

5

Apache Spark projects and demonstrations

This chapter describes projects, use cases, and worked examples illustrating how Apache Spark can be used with enterprise information on IBM z Systems platforms.

This chapter addresses the following topics:

- Emerging Apache Spark projects
- Real-time business insight with Apache Spark
- Advanced analytics of mass XML data with DB2 for z/OS and Apache Spark

5.1 Emerging Apache Spark projects

This section describes emerging projects with Apache Spark as undertaken by IBM Research.

5.1.1 Integrating Spark GraphX and IBM System G

Graph modeling and algorithms are powerful approaches for analyzing insight from linked data. In the past years, multiple technologies have emerged in applying graph for big data analytics. Such technologies include Spark GraphX and IBM System G.

Spark GraphX

GraphX is a computation system that runs in the Spark data-parallel framework for graph and graph-parallel computation. It extends Spark RDD by introducing a graph abstraction and provides a set of APIs to enable users with easy access to commonly used graph algorithms. The collection of graph algorithms and builders grows as Spark evolves. See more detail at the Spark GraphX website:

http://spark.apache.org/graphx/

IBM System G

IBM System G is comprehensive graph computing software for the big data portfolio. It is an integral full-stack solution for analytics, offering graph processing functions at all layers, such as property graph storage, run time, analytics, and visualization. System G is enabled to run on Linux on z. See more detail at the IBM System G website:

http://systemg.research.ibm.com/

IBM System G and Spark GraphX integration

To enable users to take advantage of strengths of Spark GraphX and IBM System G, research has been done to integrate Spark GraphX with System G. IBM System G is suitable for building up property-graph-based solutions, capable for both static and dynamic graphs, supporting complex analytics such as graphical models. System G now provides a plug-in to create GraphX-compatible RDDs. All property graphs stored in System G can be loaded into GraphX to enhance expressiveness in GraphX for property graphs.

5.1.2 Ease of integration and Spark with z Systems platforms

The various analytics capabilities in Spark, and its rich set of high-level APIs, position it as a viable choice for integration of data sources and business analytics functions. To enable users easily take advantage of these features in a multi-data source environment, a research project jointly developed by IBM and Tongji University aims to create a Unified Hybrid Data Extender (UHDE) framework that can be used with DB2 or other data sources on IBM z Systems platforms.

- With this framework, users can employ script-like language to define data sources, data objects, and analytics and integration operations to be applied on the data. The language maintains a syntax similar to SQL-PL and is augmented with language elements such as Spark MLlib function APIs as analytics operators.
- The input script is parsed by a parsing engine. Internally, an abstract syntax tree is generated, followed by a directed acyclic graph for optimization. A Spark job with Scala code is then generated by the framework and submitted to Spark engine for processing. If multiple data sources are referenced in the input script, the optimizer and generated

code process the request by invoking Spark's data integration APIs such as the DataFrame API.

- The framework also enables user-developed Spark functions to be defined and deployed as user-defined-operators. These user-defined-operators can be invoked similarly to other analytics operators in the UHDE's script language.
- The framework is packaged as a Java library. This lightweight integration framework work can be deployed in to Java container such as WebSphere Application Server or execute as a stand-alone Java application.
- The framework can use any relational database as its metadata store. The metadata store saves information about the hybrid execution environment such as database information, user submitted scripts, deployment information, generate execution code, and more.

An example of using UHDE framework is shown in Figure 5-1. In this example, a user-written hot-spot identification function from DB2 transaction log is deployed to a Spark system. A user script can be written to invoke the operator on a log data file, together with other data operations.

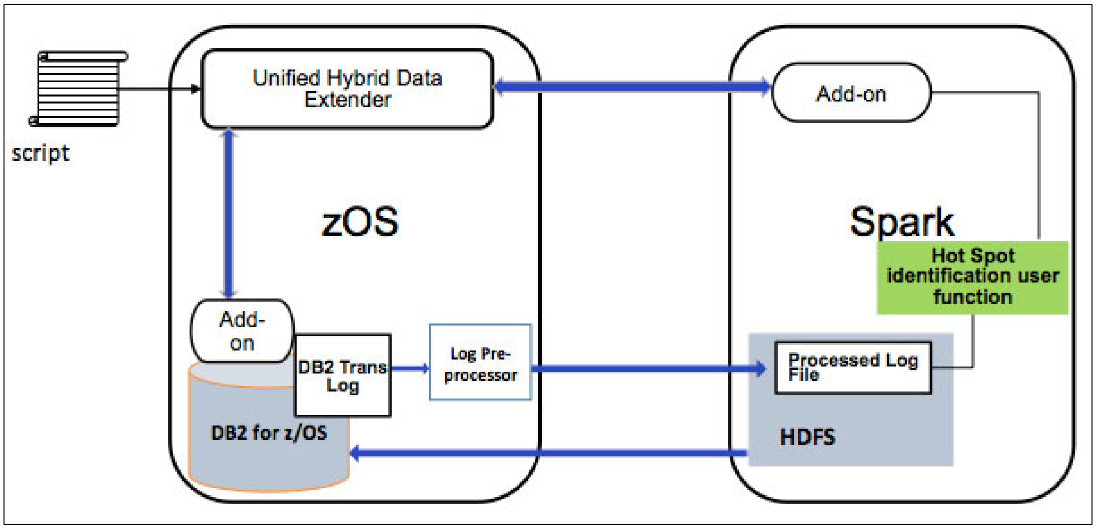


Figure 5-1 Using the UHDE framework

5.2 Real-time business insight with Apache Spark

An Apache Spark z Systems demonstration was created to highlight a business scenario using native analytics on z Systems platforms to produce real-time business insight. The purpose was to showcase the ability of Apache Spark to use both structured and unstructured data in-place as part of Spark analytics processing.

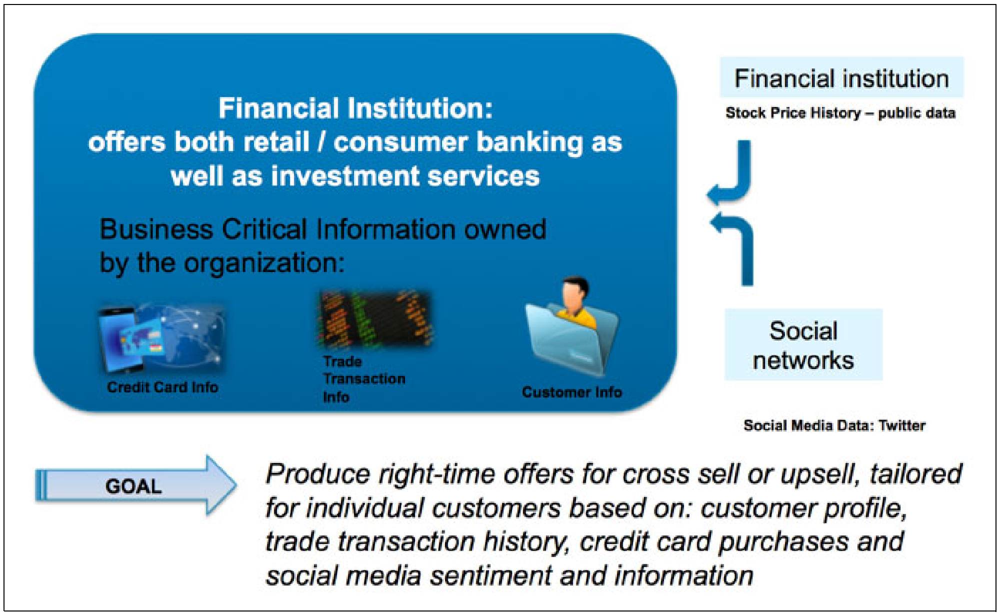
Many z Systems clients want to leverage the wealth of information they have within their OLTP environments, along with unstructured data such as that found in blogs and other social media sources. However, continuously moving all this data to a data lake or similar structure can be cumbersome, and the data is latent immediately upon moving it to a different data store. Realizing real-time benefits comes from both performing analytics quickly and in being able to use data that is most recent as part of the analytics.

5.2.1 Demonstration use case

This use case features a financial institution that offers both retail banking services and investment services. The institution has information about its clients, their credit card patterns, and their buy and sell history for stocks. The financial institution wants to be able to use this information, which they own, and combine that with external data. In this case, the external data consists of public data on stock price history from a financial services company, and also social media data found on Twitter.

The goal of the demonstration is to show how the various types of information can be analyzed in concert to produce real-time insight on cross-sell and upsell offers for this financial institution's clients, targeted and tailored for the specific individual at the right time.

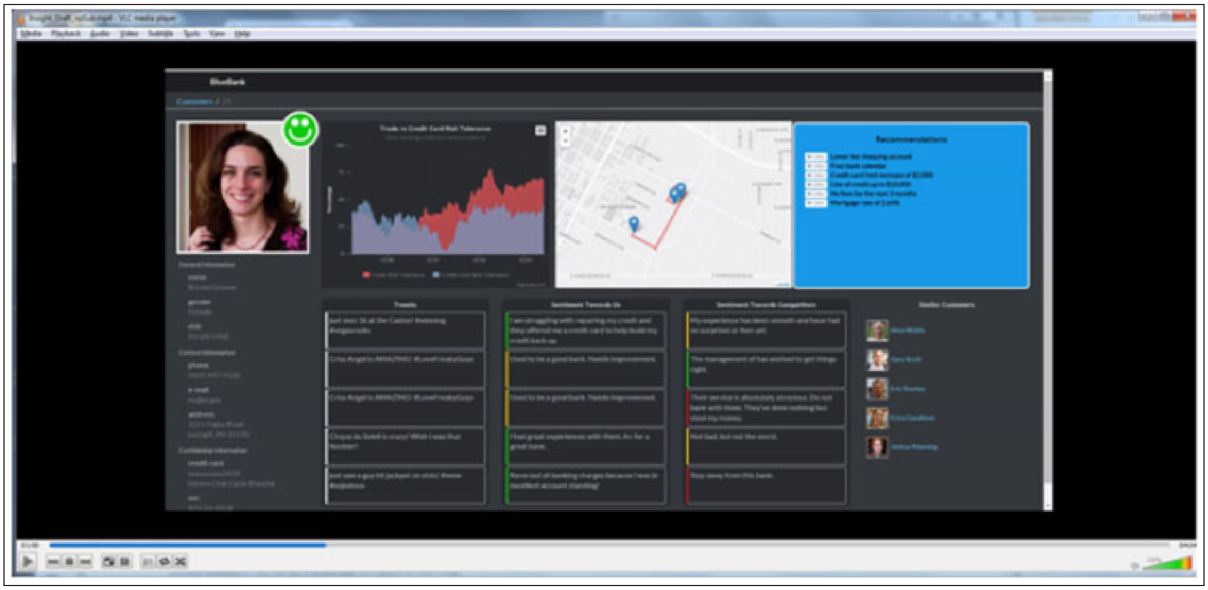
Figure 5-2 shows the business use case.



Financial Institution:
offers both retail / consumer banking as well as investment services
Business Critical Information owned by the organization:
Credit Card Info
Trade Transaction Info
Customer Info
Financial institution
Stock Price History – public data
Social networks
Social Media Data: Twitter
GOAL
Produce right-time offers for cross sell or upsell, tailored for individual customers based on: customer profile, trade transaction history, credit card purchases and social media sentiment and information

Figure 5-2 Business use case for Spark z/OS demonstration

The demonstration shows various analytics, and the results of these analytics are shown in a dashboard view as in Figure 5-3 on page 31.



Used to be a good bank. Needs improvement.

Used to be a good bank. Needs improvement.

Not bad, but not the worst.

Stay away from this bank.

Figure 5-3 Spark z/OS demonstration dashboard

For this particular customer, the dashboard shows this information:

- Profile information about the customer is under her picture; this is data that the financial institution knows about its clients.
- The graph to the right of the customer's picture changes dynamically over time and shows the customer's risk tolerance level, based on stock trades and credit card purchases. The red area shows this customer's tolerance for risk, based on her stock trades. The blue area is where this person's risk tolerance is analyzed based on her credit-card purchases. The Spark analytics can look at factors such as how quickly this customer bought and sold stocks, volatility of the stocks she purchased, the goods or services she purchased with her credit cards, and more.
- The area to the right of the graph shows the locations where this customer purchased goods or services. Clicking on one of this customer's locations will show her recent credit card transactions.
- The bottom segment of the dashboard is dynamically changing Twitter data that is classified into three columns based on general Twitter data, sentiment about the financial institution, and sentiment about competitors.
  - The left shows the customer's generic tweets that do not relate to this bank or any competitor.
  - The middle shows the changing Twitter feeds that pertain to how this customer feels about the bank (green for positive, red for negative, yellow for neutral)
  - The right shows how the customer feels about competitive banks.

  All this sentiment analysis contributes to her cumulative visual score near her picture.

The demonstration highlights the use of the organization's business-critical data on its clients, credit card purchases, and stock buy/sell information combined with sentiment data to create a list of dynamically changing sets of product recommendations. In this case, these are products and services that the financial institution can offer its clients for purposes of upsell. The suggested bank products change because what might have been a recommendation two hours ago might no longer be appropriate.

To watch this entire demonstration with narration, go to the following web address:

https://youtu.be/sDmWcu05Rk8

5.2.2 Demonstration architecture

The structure of the demonstration was constructed to use data from a variety of sources, most of which reside on z/OS, similar in profile to a large number of z Systems clients. All data about stock trades was generated by an IMS transaction system on z/OS that emulated trades, and stored that history in an IMS z/OS database. All data about credit card purchases was stored in DB2 z/OS, along with stock history information. Customer profile information was stored on VSAM on z/OS. In addition, the Twitter data was loaded into an IBM Cloudant® data environment running on Linux on z.

Figure 5-4 shows a high-level view of the architecture.

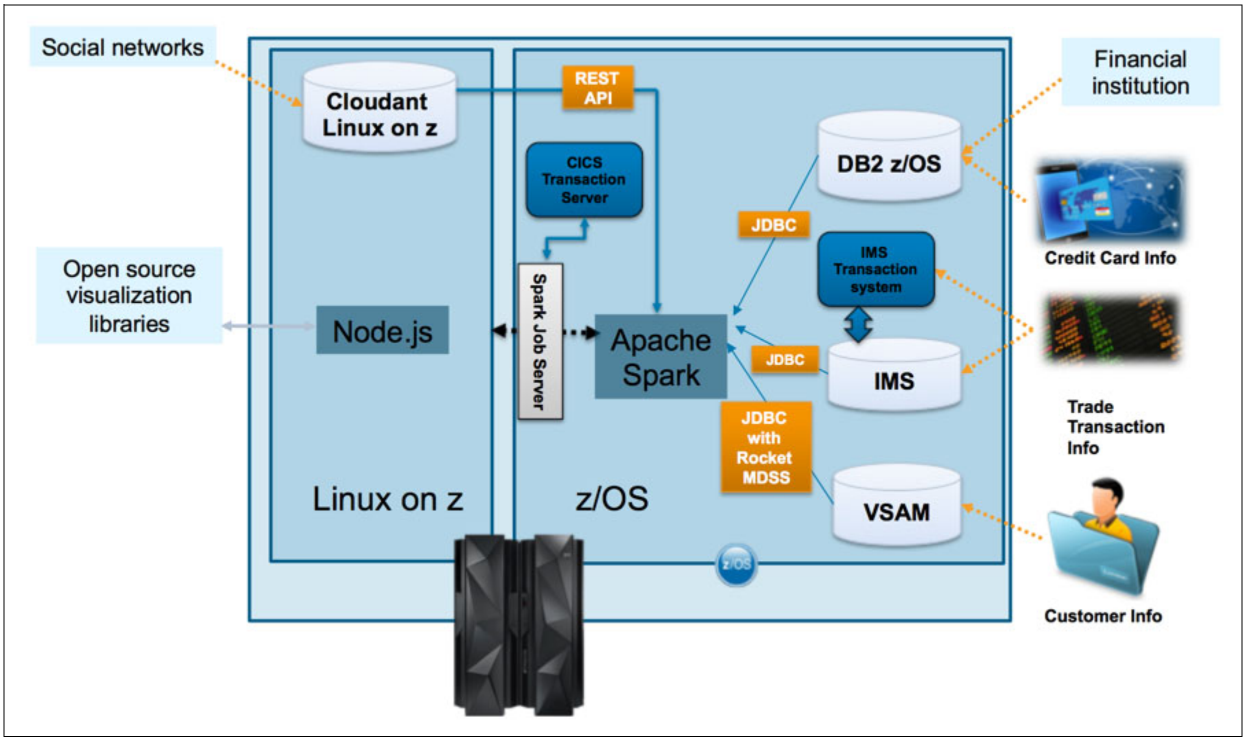


Figure 5-4 Spark z/OS demonstration architecture

In the demonstration, Spark ran natively on z/OS and accessed the various data sources without moving any of the data. For DB2 z/OS and IMS, Spark access was through JDBC drivers. For VSAM, we leveraged the optimized Rocket Mainframe Data Service for Apache Spark z/OS. For Cloudant, Spark accessed the data via REST APIs. For demonstration purposes, we also leveraged the Spark Job Server on z/OS, a separate Apache Foundation project. The Spark Job Server enabled us to create preformed Spark jobs that were invoked by the Node.js environment running on Linux on z. The Node.js layer was the interface to open source visualization libraries that were used to produce the graphics.

The example given in 4.4.1, “Invoking Spark Analytics from CICS TS” on page 24 is based on an extension to this demonstration, in which requests running in CICS Transaction Server are able to use the analytics provided by Spark in combination with real-time customer identifier information provided by the transactions to obtain a risk score.

5.2.3 Demonstration implementation

The analytics for the demonstration used various built-in capabilities of Spark, including sentiment analysis such as bag of words and recursive deep models, aggregations, natural joins, sorting, and others. The analytics code was written in Scala.

Spark running on z/OS was selected as a core part of the demonstration environment to highlight the value of co-location with the data and transaction environments. Spark z/OS will read data from DB2 z/OS, IMS, VSAM as well as Cloudant into RDD structures residing in memory on z/OS. The processing is kept local and therefore more optimal in terms of performance and security.

5.3 Advanced analytics of mass XML data with DB2 for z/OS and Apache Spark

Spark can be used in conjunction with DB2 for z/OS to perform analytical tasks on vast amounts of data without impacting transactional performance. This section examines a scenario for how to provide this capability for users that run XML analytics using the Spark framework with Spark SQL. The concept makes the use of the in-memory clustered computing without storing the data outside of an existing database.

This is a proof-of-concept prototype; it showcases that heavy-lifting analytical workload outside of the relational database. The XML data is transferred from DB2 for z/OS into Spark memory structures, enabling fast computing and advanced analytics by using the Spark system.

5.3.1 Background

DB2 is a powerful database product designed for storing, analyzing, and retrieving data efficiently. However, it is sometimes not the preferable solution for running analytics applications against massive amounts of data, requiring repetitive large table scans. A new modernized technology for analytics and machine learning can mitigate the challenges as cost-friendly as possible.

This project was initiated to address a requirement shared by a number of IBM DB2 for z/OS customers globally. Today, XML data can be manipulated using script language, such as XQuery, xPath, and others. A DB2 for z/OS user can write query expressions to navigate through XML's hierarchical data structures, and receive sequences of XML documents. But streaming real time analytics becomes more pervasive, and there will be more demand to process data in XML documents using analytical query processing support.

5.3.2 Use case scenario

This is an example from a financial institution using DB2 queries on 400 TB of profile information of tax payers and that is stored as XML documents within the database. The XML data can be analyzed interactively by the user through online analytics processing tools from multiple dimensions.

The main goal of this use case is to enable advanced analytics on this business data without impacting the existing online transactional processing (OLTP). Such analytics allow data scientists to run queries on XML data that originates on the mainframe for various purposes such as IT monitoring, fraud detection, hotspot analysis, and more.

This DB2 use case introduces a number of challenging issues. From a system's perspective, no technology is available to analyze data in XML format that uses a real-time query engine such as Hadoop Distributed File System (HDFS). Second, a large scale data offloading process from a DB2 XML table to Hadoop clusters is a performance bottleneck and can impose a complete new set of data currency issues if the XML must be maintained continuously. Furthermore, DB2 has support for XML manipulation and functions for parsing. This support is effective for transactional processing, such as finding specific sub-nodes in an XML database. However, they are not designed for XML analytics, such as shredding the data and running analytical applications or tooling.

Processing large amounts of XML data offers challenges in DB2 for z/OS today. Parsing XML on the z/OS platform might use high CPU consumption and be potentially impractical for very large documents. In addition, it might consume significant amounts of memory and potentially result in storage issues. The problem becomes more severe if the XML document is large and deeply nested or has many optional fields.

Addressing these requirements is key to success for DB2 in terms of enabling large scale XML analytical support. Starting from DB2 for z/OS Version 10, the OLAP capability on the mainframe is generally supported by the IBM DB2 Analytics Accelerator. However, the XML data type is currently not supported in this combination, which requires a solution outside of this well-known accelerator.

5.3.3 Solution

The solution to this concrete business problem is to incorporate Apache Spark with DB2 for processing mainframe-hosted XML data by developing a Spark application. The overall design enables fast data pipelining from DB2 into a Spark RDD. After the data is loaded in Spark as collections of data partitions, transform the XML data for aggregation and analytics. The purpose of doing so is to push heavy lifting analytical processing workloads outside of DB2 to reduce CPU cost and enable new machine learning capabilities by using the Spark framework. Data scientists can then query the data by scanning the entire XML table in DataFrame format in which now complex queries can run faster and more efficiently than before, while retaining the transactional lookup performance for specific XML nodes of the DB2 engine.

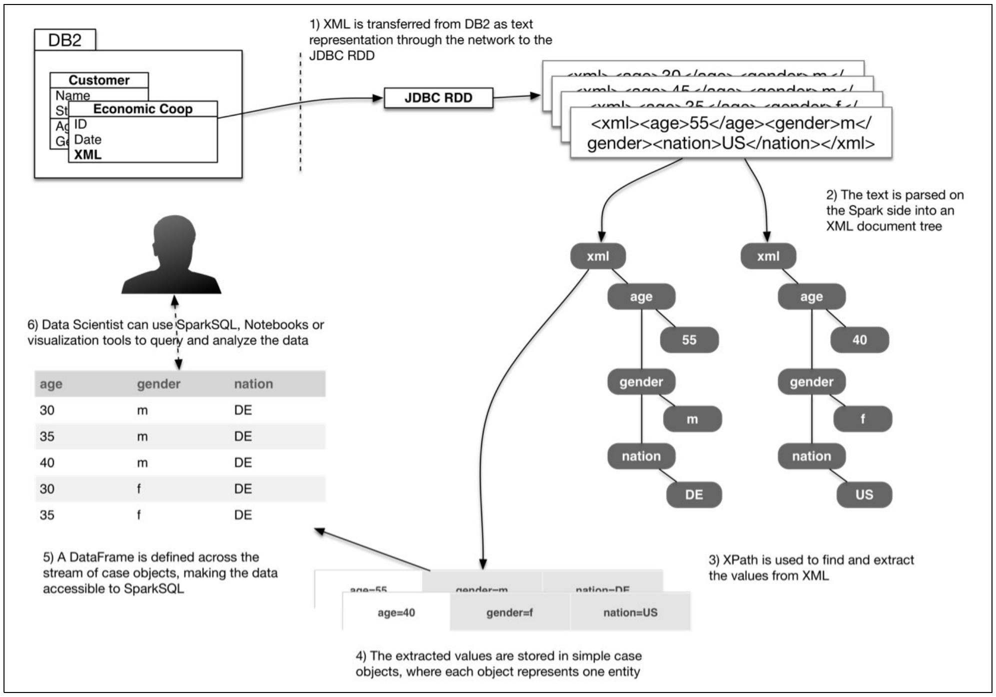
5.3.4 Apache Spark on DB2 for z/OS data

To achieve this goal, the middleware support is given by data streaming from DB2 through a Spark API named JDBCRDD. This function call in Java is used to load data from DB2 database through a JDBC connection driver. The data is kept in Spark memory as needed, instead of being persisted on HDFS. This function call creates multiple tasks for JDBC connections, each executing sub-queries for a portion of the source table data. The number of tasks is based on the Spark optimizer by using a key range. As a result, data is streamed into Spark memory as parallel data scans from different DB2 partitions of the XML table.

Apache Spark supports building stand-alone applications in Scala, Java, and other programming languages for advanced analysis. The following steps show the flow of the application being processed (see Figure 5-5 on page 35):

1. Load Data Frame with JDBC data source where the XML is received as a string (CLOB).
2. Parse the XML in Scala `xml` document format.
3. Load all records from high-level XML nodes where values of all nodes are normalized and mapped to a record/case class. This is essentially mapping the unstructured XML data to structured records of values that are interesting for later analytical processing.

4. Register the DataFrame as a temporary table.
5. Optional: Cache the registered table.
6. Run queries using Spark SQL or use ML algorithms to detect correlations.



1) XML is transferred from DB2 as text representation through the network to the JDBC RDD

2) The text is parsed on the Spark side into an XML document tree

3) XPath is used to find and extract the values from XML

4) The extracted values are stored in simple case objects, where each object represents one entity

5) A DataFrame is defined across the stream of case objects, making the data accessible to SparkSQL

6) Data Scientist can use SparkSQL, Notebooks or visualization tools to query and analyze the data

age	gender	nation
30	m	DE
35	m	DE
40	m	DE
30	f	DE
35	f	DE

Figure 5-5 Application flow

The following sample code is used to demonstrate the feasibility of the Spark application. The source data that is processed in this example is retrieved from the Organization for Economic Cooperation and Development. The content of data is matrices showing employment rates of European countries on a quarterly basis, sorted by gender. This data was loaded into a DB2 for z/OS table by using the XML data type. The initial data is pipelined into Spark memory as shown in the examples.

1. Connect to DB2 through JDBC:
   a. Construct a JDBC URL (Example 5-1).

Example 5-1 Construct JDBC URL

```
val url="jdbc:db2://theDB2Server:446/DBLOC"
url: String jdbc:db2://theDB2Server:446/DBLOC
```

   b. Create a connection properties object with user name and password (Example 5-2).

Example 5-2 Connection properties

```
val prop = new java.util.Properties
prop: java.util.Properties = {}

prop.setProperty("user","db2user")
prop.setProperty("password","secret")

res158: Object = null
res159: Object = null
```

2. Load XML as Strings. Get a DataFrame with JDBC data source (`url`, `table_name`, `connection_properties`). This is shown in Example 5-3, along with sample output.

Example 5-3 DataFrame

```
val xmlRecord = sqlContext.read.jdbc(url,"tbdk157a",prop)
xmlRecord: org.apache.spark.sql.DataFrame = [ID1: int, MYXML1: string]
+---+--------------------+
|ID1|              MYXML1|
+---+--------------------+
|  1|<DataSet keyFamil...|
|  2|<DataSet keyFamil...|
+---+--------------------+
```

3. Create a case class to store the XML attribute values in relational form (Example 5-4).

Example 5-4 Case class

```
case class Record(countryID: String, gender: String, age: String, time:String,
obsValue: Double)
defined class Record
```

4. Parse XML into a document (Example 5-5).

Example 5-5 Parse XML into document

```
val xmleveryDoc = xmlDoc.collect.map( _.getString( 0 ) ).map( scala.xml.XML.loadString(
_ ) )
xmleveryDoc: Array[scala.xml.Elem] = Array(
<DataSetkeyFamilyURI="http://stats.oecd.org/RestSDMX/sdmx.ashx/GetKeyFamily/GENDER_EMP">
<KeyFamilyRef>GENDER_EMP</KeyFamilyRef>
<Series>
   <SeriesKey>
      <Value value="AUS" concept="COU"/>
      <Value value="EMP4_E" concept="IND"/>
      <Value value="MEN" concept="SEX"/>
      <Value value="15PLUS" concept="AGE"/>
      <Value value="Q1-2007" concept="TIME"/>
   </SeriesKey>
   <Attributes>
      <Value value="Percentage" concept="UNIT"/>
      <Value value="0" concept="POWERCODE"/>
   </Attributes>
   <Obs>
      <ObsValue value="69.3"/>
   </Obs>
</Series>
<Series>
   <SeriesKey>
      <Value value="AUS" concept="COU"/>
      <Value value="EMP4_E" concept="IND"/>
      <Value value="MEN" concept="SEX"/>
      <Value value="15PLUS" concept="AGE"/>
      <Value value="Q1-2008" concept="TIME"/>
   </SeriesKey>
   <Attributes>
      <Value value="Percentage" concept="UNIT"/...\
```

5. Load all records from Series node where all values of nodes are normalized and mapped to the Record class. Here only the Country, Gender, Age, Time, and employment rate value are selected as part of the case class attributes. Note the following mapping scheme:

   ```
   s => s \\ "@value"
   ```

   It interprets that the values are always in the same order, therefore the transformation can always be easily mapped to array n for the pseudo class. See Example 5-6.

   Example 5-6 allSeriesRecord

   ```
   val allSeriesRecord = (xmleveryDoc(0) \\ "Series").map(s => s \\ "@value"
   ).toArray.map{ n => Record(n(0).text, n(2).text, n(3).text, n(4).text,
   n(7).text.toDouble) }
   allSeriesRecord: Array[Record] = Array(Record(AUS,MEN,15PLUS,Q1-2007,69.3),
   Record(AUS,MEN,15PLUS,Q1-2008,69.9), Record(AUS,MEN,15PLUS,Q1-2009,68.6),
   Record(AUS,MEN,15PLUS,Q1-2010,68.4), Record(AUS,MEN,15PLUS,Q1-2011,69.1),
   Record(AUS,MEN,15PLUS,Q1-2012,68.3), Record(AUS,MEN,15PLUS,Q1-2013,67.7),
   Record(AUS,MEN,15PLUS,Q2-2007,69.7), Record(AUS,MEN,15PLUS,Q2-2008,69.8),
   Record(AUS,MEN,15PLUS,Q2-2009,68.1), Record(AUS,MEN,15PLUS,Q2-2010,68.4),
   Record(AUS,MEN,15PLUS,Q2-2011,68.6), Record(AUS,MEN,15PLUS,Q2-2012,68.2),
   Record(AUS,MEN,15PLUS,Q2-2013,67.5), Record(AUS,MEN,15PLUS,Q3-2007,69.7),
   Record(AUS,MEN,15PLUS,Q3-2008,69.8), Record(AUS,MEN,15PLUS,Q3-2009,68.0),
   Record(AUS,MEN,15PLUS,Q3-2010,68.8), Record(AUS,MEN,15PLUS,Q3-2011,68.4),
   Record(AUS,MEN,15PLUS,Q3-2012,67.8), Record(AUS,MEN,15P...
   ```

6. Make the data accessible as a query:

   a. Create an RDD for all the records and convert it into DataFrame (Example 5-7).

      Example 5-7 Create RDD

      ```
      val allrecordDF = sc.parallelize(allSeriesRecord ).toDF()
      allrecordDF: org.apache.spark.sql.DataFrame = [countryID: string, gender:
      string, age: string, time: string, obsValue: double]
      ```

   b. Register this DataFrame as a temporary table named `"recordtable"` (Example 5-8).

      Example 5-8 Register DataFrame

      ```
      allrecordDF.registerTempTable("recordtable")
      ```

   c. Optional: Cache the table data (Example 5-9).

      Example 5-9 Clear table cache

      ```
      allrecordDF.cache
      res182: allrecordDF.type = [countryID: string, gender: string, age: string,
      time: string, obsValue: double]
      ```

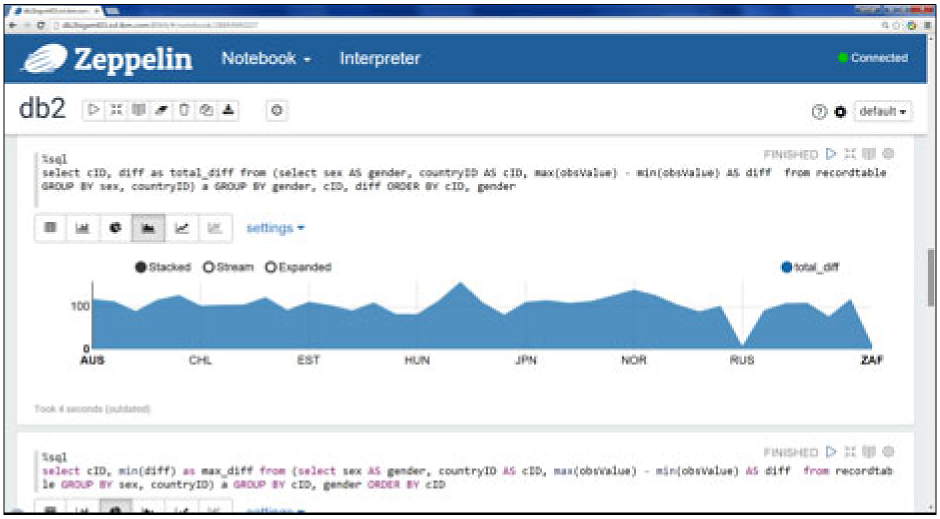
7. Generate a sample query. An example to select average of employment rate for all countries for only men is shown in Example 5-10.

   Example 5-10 Sample query

   ```
   sql ("select avg(obsValue) from recordtable where
   gender='MEN'").collect.foreach(r => println("gender: MEN " + " rate= " + r(0)))
   gender: MEN  rate= 35.00280561122244
   ```

5.3.5 Data visualization with Apache Spark

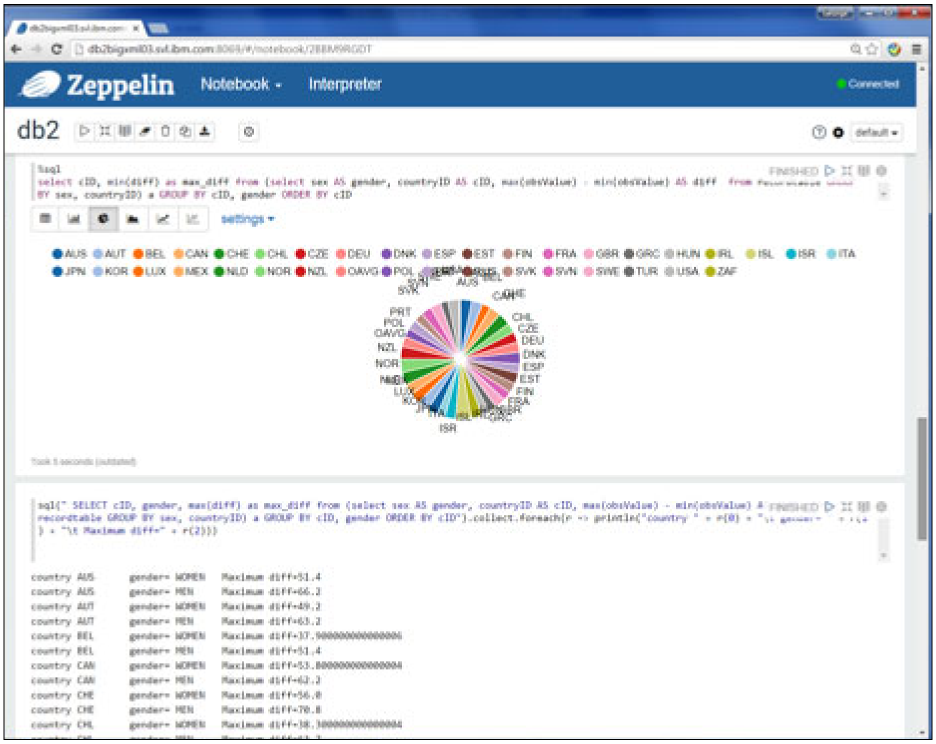
After data is loaded in Spark memory, the XML data is transformed to the normalized records as defined in a case class structure. Apache Zeppelin is used for this project to visualize data. This allows data scientists to perform data exploration on a cloud-based platform. As part of an open source effort, Zeppelin is also supported by the IBM Open Platform, having many language interpreters such as Scala, Spark SQL, and others. In the example, a query is issued in Spark SQL to find out the different employment rates of each country by gender. The result is presented in graphical charts Figure 5-6 and Figure 5-7.



```
%sql
select cID, diff as total_diff from (select sex AS gender, countryID AS cID, max(obsValue) - min(obsValue) AS diff  from recordtable
GROUP BY sex, countryID) a GROUP BY gender, cID, diff ORDER BY cID, gender
```

```
%sql
select cID, min(diff) as max_diff from (select sex AS gender, countryID AS cID, max(obsValue) - min(obsValue) AS diff  from recordtab
le GROUP BY sex, countryID) a GROUP BY cID, gender ORDER BY cID
```

Figure 5-6   Data visualization



```
%sql
select cID, min(diff) as max_diff from (select sex AS gender, countryID AS cID, max(obsValue) - min(obsValue) AS diff  from recordtable GROUP
BY sex, countryID) a GROUP BY cID, gender ORDER BY cID
```

```
sql(" SELECT cID, gender, max(diff) as max_diff from (select sex AS gender, countryID AS cID, max(obsValue) - min(obsValue) A
recordtable GROUP BY sex, countryID) a GROUP BY cID, gender ORDER BY cID").collect.foreach(r => println("country " + r(0) + "\t gender="
) + "\t Maximum diff=" + r(2)))
```

```
country AUS     gender= WOMEN   Maximum diff=51.4
country AUS     gender= MEN     Maximum diff=66.2
country AUT     gender= WOMEN   Maximum diff=49.2
country AUT     gender= MEN     Maximum diff=63.2
country BEL     gender= WOMEN   Maximum diff=37.900000000000006
country BEL     gender= MEN     Maximum diff=51.4
country CAN     gender= WOMEN   Maximum diff=53.800000000000004
country CAN     gender= MEN     Maximum diff=62.2
country CHE     gender= WOMEN   Maximum diff=56.0
country CHE     gender= MEN     Maximum diff=70.8
country CHL     gender= WOMEN   Maximum diff=38.300000000000004
```

Figure 5-7   Data visualization

5.3.6 Conclusion

DB2 XML analytics is extended by incorporating Apache Spark technology and pipelining the raw data from a DB2 XML table into a Spark RDD. This way saves CPU costs compared to direct query execution against XML documents within from DB2. The only CPU cost on the mainframe side is associated with the data pipelining when reading the XML table. Such a pipelining process is required once to query the entire XML table. After the XML documents are converted to the normalized data form, data aggregation and data analytics can be performed. Because the data parsing and transformation is done on the Spark side, memory consumption and storage issues are less of a concern. The use of Spark SQL with cached DataFrames allows repeated querying to increase the efficiency of the process. With the incorporation of the Spark framework, a whole new machine learning capability is extended to DB2 for mass XML documents analytics in addition to the existing extraordinary built-in XML capability of DB2 for processing single documents.

The next step is to enhance the ad hoc analytical processing application with the ability to send results back from Spark to the DB2 database.

REDP-5336-00

ISBN 0738455040

Printed in U.S.A.