Whitepaper | Supercharging BI with Spark

A Little Bit of History

Ever since the creation of the Apache Hadoop project more than 10 years ago, many attempts have been made to adapt it for data visualization and analysis. The original Hadoop project consisted of two main components, the MapReduce computation framework and the HDFS distributed file system. Other projects based on the Hadoop platform soon followed. The most notable was Apache Hive which added a relational database-like layer on Hadoop. Together with a JDBC driver, it had the potential to turn Hadoop into a Big Data solution for data analysis applications.

Unfortunately, MapReduce was designed as a batch system, where communication between cluster nodes was based on files, job scheduling was geared towards batch jobs, and latency of up to a few minutes is quite acceptable. Since Hive used MapReduce as the query execution layer, it was not a viable solution for interactive analytics, where sub-second response time is required.

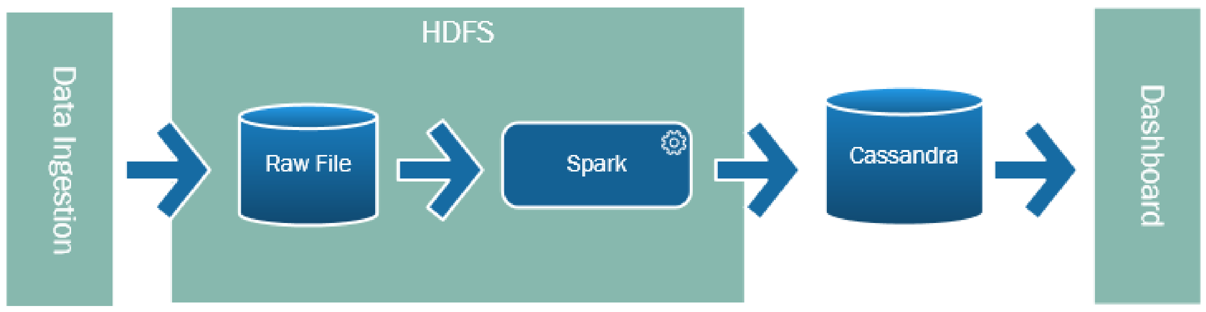
This didn't change until Apache Spark came along. Instead of using the traditional MapReduce, Spark introduced a new real-time distributed computing framework. Furthermore, it performs executions in-memory so job latency is much reduced. In the same timeframe, a few similar projects have emerged under the Apache Hadoop umbrella such as Tez, Flink, and Apex. Finally, interactive analysis of Big Data was within reach.

Current State of Art

As Spark quickly gained the status of the leading real-time cluster framework, it has attracted much attention in the BI community. By now, almost every BI vendor has some kind of story about integrating their products with Spark.

The most common integration is to treat Spark as a new data source. A BI tool could be connected to a Spark/Hadoop cluster through a JDBC/ODBC driver. Since Spark SQL provides an SQL-like language, it could be treated as a traditional relational database. This approach is simple and easy to accomplish, so it is normally the first option when a BI tool integrates with Spark.

Some tools tried to go further. One option is to use Spark as a replacement for ETL. Instead of the traditional ETL pipeline, data could be ingested into a Hadoop cluster, and Spark is used to transform and process the raw data. The result is saved into another data store to be consumed by the BI tool.

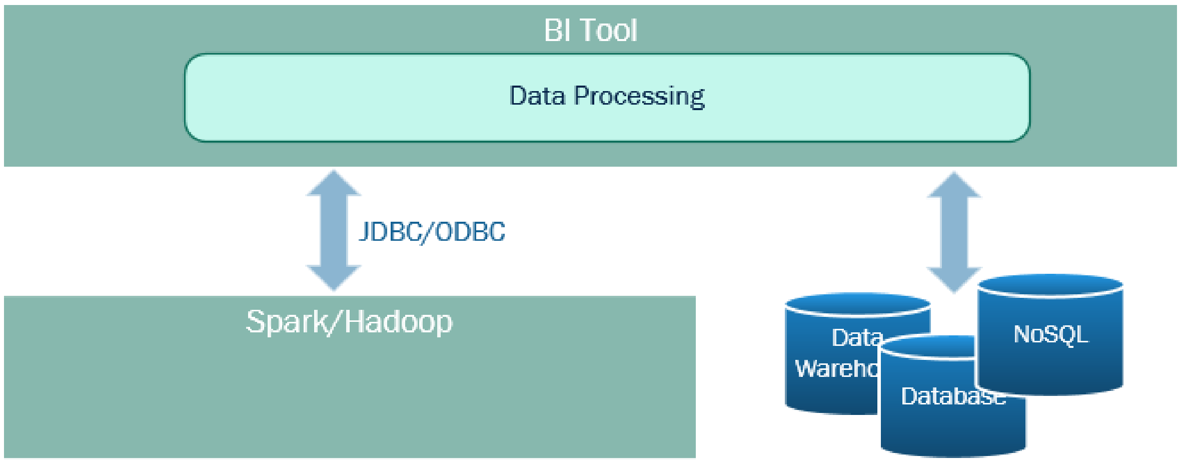


Native Spark Integration

Using Spark in various roles outlined above has its value, and Spark will continue to be an important part of the overall BI pipeline. However, keeping Spark outside of the BI tool fails to take full advantage of the power of Spark in many ways.

First, having Spark as an external system means data needs to be moved from a Spark cluster to the BI tool. In the age where software is actively engineered to reduce data movement even between RAM and CPU cache, moving data between machines or processes can be disastrous to performance.

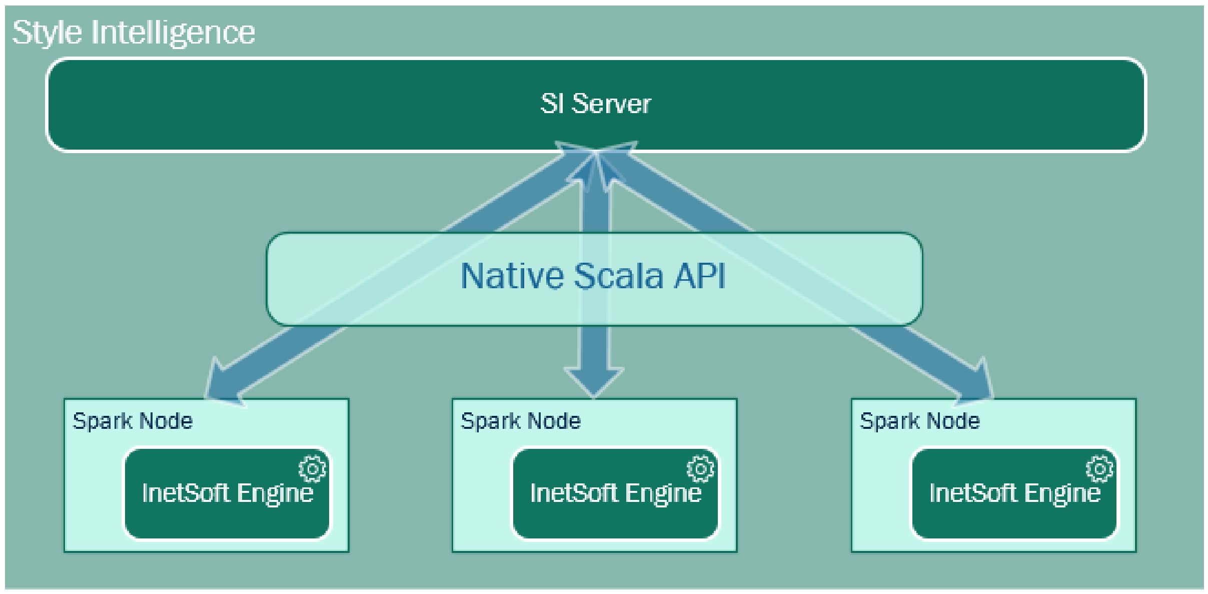
Secondly, treating Spark simply as a database, or even as a preprocessor of data, robs a BI tool of the opportunity to fully utilize the computing power of a cluster. Imagine when you need to join data from Spark and a simple in-memory reference table. Since the in-memory table is not part of the Spark cluster, the BI tool needs to execute a query against Spark, bring the data out of the cluster and into the BI tool, and then perform a join in the BI tool.



This example illustrates the two main disadvantages of keeping a BI tool separate from Spark. First, potentially large amounts of data may need to be moved between systems. Additionally, joining of the data cannot be performed inside the cluster. This may result in the cluster being idle while a single BI server is consumed with performing the actual data processing.

The InetSoft Approach

To avoid the aforementioned problems, the key is to break down the barrier between Spark and the BI tool. Instead of relying on the JDBC/ODBC interface, we completely fused Spark into Style Intelligence. The following is a high-level view of the integration.



All interactions between Style Intelligence and Spark are through the native API. Spark becomes a native part of the product. Data inside a cluster can be accessed through Spark SQL using SQL-like queries, or directly as files or data connectors provided by the data stores. Unlike JDBC, data is not retrieved from Spark until it has finished all processing and is ready to be presented to users.

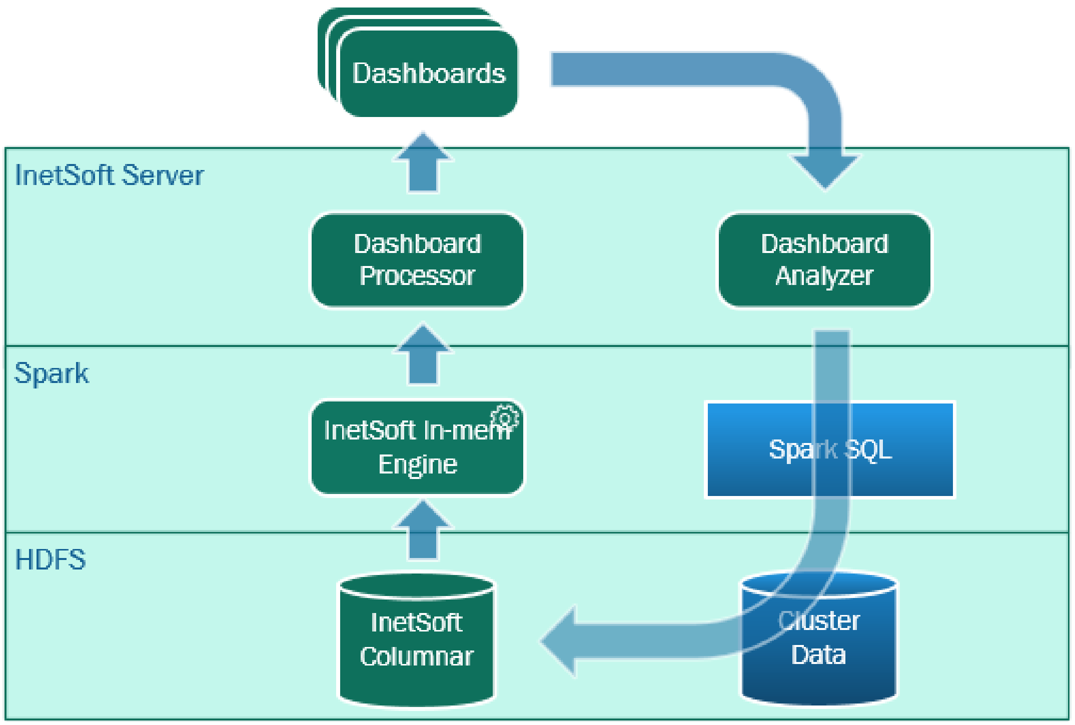
Equally importantly, data processing jobs normally performed by BI tools are also pushed into the cluster. For example, to join the result of a Spark SQL query with an in-memory table, the in-memory table is pushed into the cluster, and the join is submitted to the InetSoft execution engine embedded in the Spark nodes. Data already residing in the cluster is never moved.

Analytic Query Acceleration

Spark is a general purpose computing platform. Although it was designed to handle real-time computing needs, it provides no special treatment for analytic queries. A BI tool could use Spark as-is and get decent performance. But BI queries generated by interactive analysis often have distinct patterns, and it is possible to use this knowledge to further optimize the execution.

As a natively integrated tool, Style Intelligence is in a unique position to add special logic to improve performance further. The acceleration is achieved by three methods:

1. Style Intelligence analyzes a visualization and generates candidate queries to materialize. The goal is to preprocess as much as possible and only leave the parts that need to be dynamically updated to be performed later.
2. For data stored in a materialized format, logic responsible for handling interactive queries is pushed down to the data storage layer whenever possible. This significantly reduces the amount of data Spark needs to process.
3. A specialized columnar data store is created to store the materialized data. It conforms to the standard Hadoop/Spark API so it can be accessed by Spark jobs, but is optimized to handle the types of queries generated from interactive analysis.



Spark Is More Than Big Data

While Spark gained its fame through its real-time cluster computing platform, it comprises many other components. Many of those are of great interest to BI users. Ignoring them would be a great disservice to users. Chief among them are Machine Learning (Spark ML) and streaming. We consider both areas as core capacities of any future BI tool and have brought them into the product as part of the native integration with Spark.

Spark ML

Spark ML provides an opportunity to bring Machine Learning to the masses, and we view Style Intelligence as the bridge from Spark ML to business users. The integration has the following objectives:

1. Creating and training models should be easy and accessible to people with minimum Machine Learning training. Every opportunity was taken to automate the process.
2. Once a model is trained, it should be available to business users with no Machine Learning knowledge.
3. Like Spark queries, Spark ML integration is done at the native level, fully integrated into the product, so it enjoys all the benefits of the cluster.

Spark ML provides built-in algorithms in the following areas:

1. Classification – prediction of categorical values
2. Regression – prediction of numeric values
3. Clustering – automatic grouping of similar items
4. Recommendation

The first three are available in Style Intelligence and can be accessed as a regular query without any programming or special knowledge.

Spark Streaming

Streaming is another area that is revolutionizing BI applications. Not long ago, real-time data processing was considered a niche requirement and often reserved for the most specialized applications. With the advent of Kafka and Spark, streaming has been brought into the mainstream.

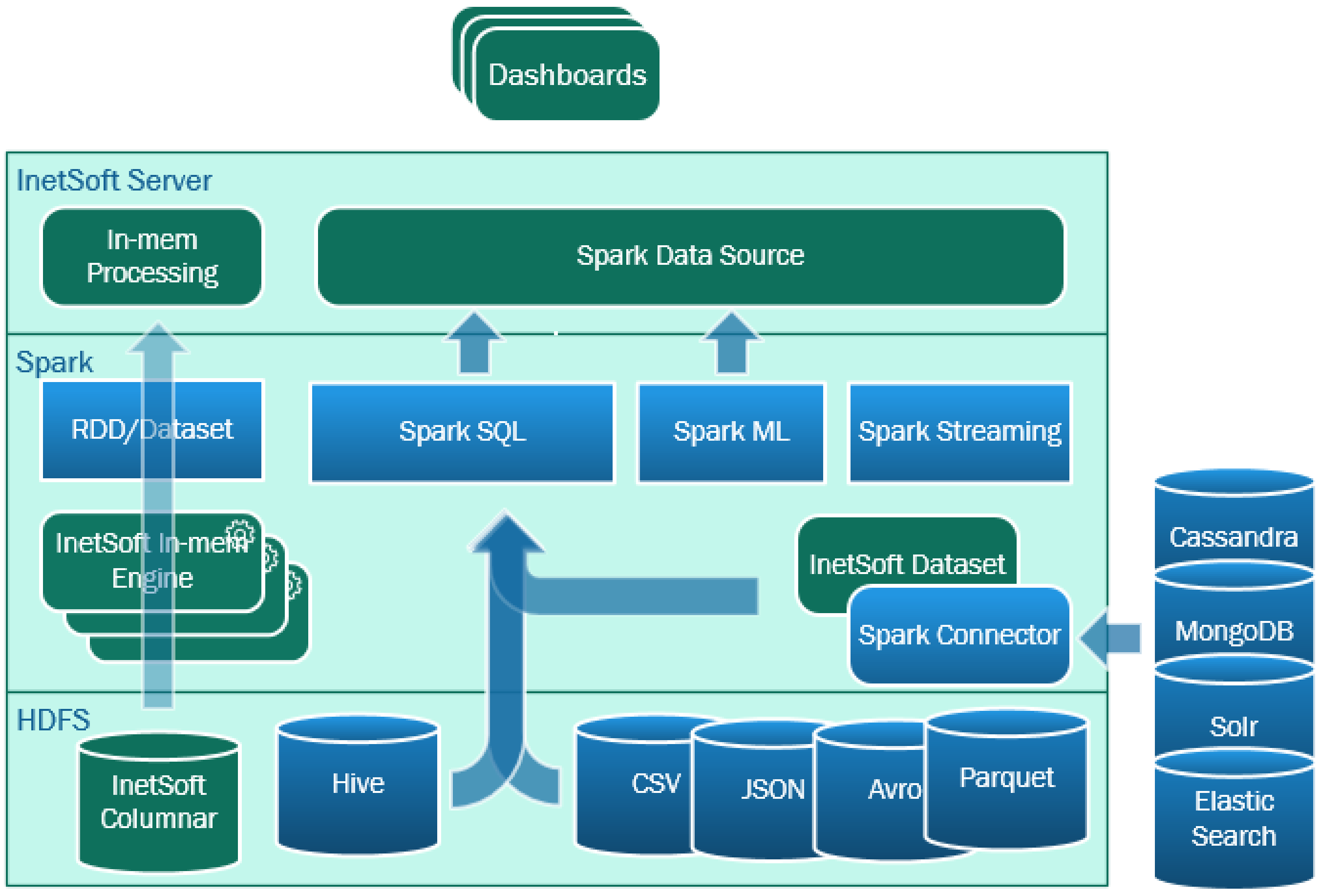
However, many challenges remain. Compared with batch processing, streaming needs to deal with many new problems such as back pressure and late arriving records. In addition, programming is often required to create a streaming application.

Spark Streaming provides an abstraction of stream processing that nicely fits into the overall Spark framework. We are actively working on integrating streaming into Style Intelligence. The goal is to simplify stream processing and make it accessible to regular users without programming skills. To a large degree, building a query against a stream should not be much different from a query against a database.

Summary

Big Data is more than throwing a cluster together and connecting to it through a JDBC driver. It requires an architecture that has all the pieces working closely together. It's our view that the most effective solution requires a complete fusion of the various technologies.

Style Intelligence/Spark integration is the first step in achieving this vision. As the innovations flourish, our mission is to create a platform that effortlessly integrates with software in the Hadoop ecosystem, so business users can enjoy all the benefits.


Dashboards
InetSoft Server
In-mem Processing
Spark Data Source
Spark
RDD/Dataset
Spark SQL
Spark ML
Spark Streaming
InetSoft In-mem Engine
InetSoft Dataset
Spark Connector
HDFS
InetSoft Columnar
Hive
CSV
JSON
Avro
Parquet
Cassandra
MongoDB
Solr
Elastic Search