A Report on Apache Apex

SRIKANTH RAMANAM[1]

[1] *School of Informatics and Computing, Bloomington, IN 47408, U.S.A.*
[*] *Corresponding authors: srikrama@iu.edu*

March 27, 2017

Apache Apex is a Hadoop YARN native big data processing platform with both stream and batch processing capabilities. This paper explores the architecture, functioning and competition of Apache Apex.

Keywords: Stream, Processing, YARN, Apache, Apex, Malhar, I524

https://github.com/cloudmesh/sp17-i524/raw/master/paper2/S17-IR-2028/report-review.pdf

INTRODUCTION

Apache Apex was developed to address the enterprises need to process and analyze real time data stream. It was initially developed by DataTorrent as the core engine for their RTS platform, a platform for processing, analysis and visualization of stream data. DataTorrent later decided to make it open source and submitted the proposal for Apache Apex to Apache incubator in 2015 [1]. Apache Apex was accepted to the Apache Incubator and later several enterprises CapitalOne, DirecTV, General Electric, Apple and Silver Spring Networks joined its open source community. Apache Apex was first released in 2016. Apache Apex is built over YARN and is compatible with existing Hadoop platforms.

COMPONENTS

Apex has two main components [2].

Apex Core

Apex Core is the framework for building distributed applications on Hadoop.

Apex Malhar

Malhar provides a library of operators. They are mainly of two types

Input/Output Operators

Input/Output operators: These operators offer connectivity with a variety of existing data sources.

Compute Operators

Compute Operators: These operators offer functionality of Machine Learning, Stats and Math, Pattern Marching, Query and Scripting, Stream manipulators, Parsers and UI & Charting.

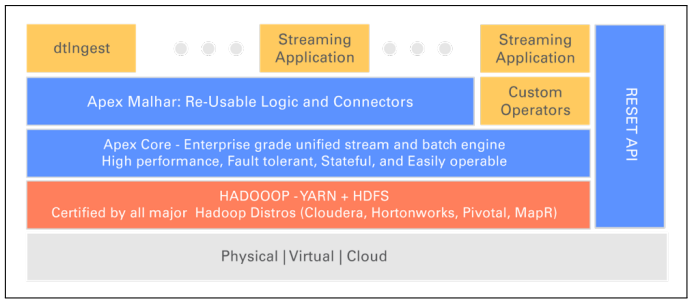

Fig. 1. Apache Apex Components [3]

ARCHITECTURE

Operators are the basic blocks of Apex applications. A streaming application is built using in-built or custom operators are connected to form a DAG (Directed Acyclic Graph) using streams.

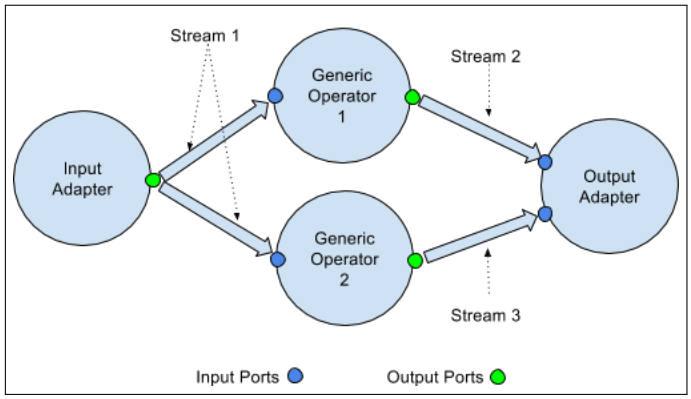

Fig. 2. Apex Application DAG [4]

APPLICATION DEVELOPMENT

Apex applications can be written in Java using any IDE supporting Java like Eclipse. Other prerequisites include Apache Maven 3.0., Apache Apex, Apache Malhar [4].

Operators

Operators are independent units of logical operations that either contribute to a part of or a whole business use case. An operator has an input port to receive data tuples and an output port to send data tuples to another operator or external system [5]. Types of Operators [5]:

- Input Adapter: An operator at the beginning of the DAG to receive data from an external system.
- Generic Operator: Accepts tuples from previous operator in DAG and does some processing task and outputs the processed data to another operator.
- An operator at the end of a DAG and outputs the data tuples to an external system.

API [5]:

- setup() initializes the operator.
- process() performs the core processing operations on data tuples and gets triggered when tuples are received.
- beginWindow() and endWindow() are used for pre and post processing steps.
- teardown() shuts down the operator and releases the resources held by the operator.

Directed Acyclic Graph

A DAG is constructed to accomplish a business task using several operators connected through streams [4].

Constructing a DAG:

- Operators are added to a DAG using dag.addOperator(args) [6].
- Streams are added to a DAG using dag.addStream(args) [6].
- Other configurations related to YARN can also be added to DAG [6].

Package

Apex applications are assembled and shared using Apache Apex Packages, which are zip files with all necessary files to launch those applications. Apache Apex Packages are created using Maven. First a Maven project is created with path to the application code. Then a mvn package command creates an application package in the target package.

Zip structure of a mvn package:

- app contains jar files of the DAG code
- lib contains jar files of dependencies
- conf contains preset configuration
- META-INF contains meta information in files MANIFEST.MF and properties.xml
- resources for other files

FUNCTIONING

Apex applications are usually packaged, shared and are usually deployed over Hadoop clusters. It is also compatible with several popular data sources as shown in the below figure.

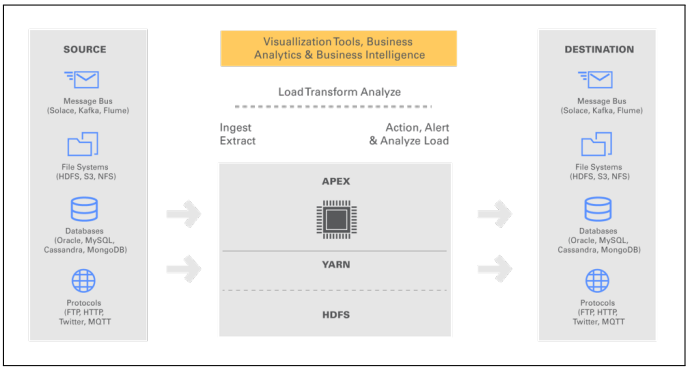

Fig. 3. Apache Apex Interoperability[3]

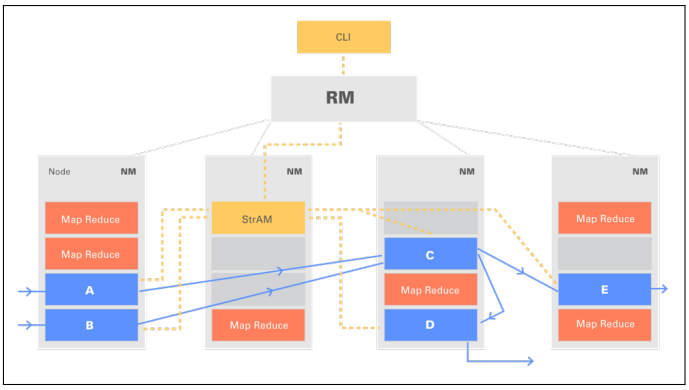

Fig. 4. Apache Apex Application Example [3]

USER INTERFACE

A command line interface called Apex CLI is available for Apache Apex. It can be launched with command apex. help command is available for all commands to obtain information and syntax. Real-time visualization dashboards can be plugged into applications for visual analytics.

LICENSE

Apache Apex is licensed under Apache License 2.0 [7].

PRICING

Apache Apex is a free open source software.

COMPETITION

Some of the competitors of Apache Apex for stream processing and analytics are [8]:

Apache Spark

This is a large-scale data processing engine that also offers stream processing [9]. But this is not a pure streaming engine as it accomplishes the same through micro-batching, fast execution of batches on small sets of data.

Apache Flink

This is an open-source stream processing framework that processes streams in real-time [10]. Almost similar to Apex but not as widely used.

Apache Storm

This is a free and open source distributed real-time computation system [11]. This is fast but not stateful like Apex.

Apache Samza

This is a distributed stream processing framework [12]. It was first developed by LinkedIn and later opensourced.

USERS

Enterprises like GE, PubMatic, SilverSpring Networks are using Apex based streaming solutions.

CONCLUSION

Apache Apex is an open source YARN(Hadoop 2.0)-native platform [2]. It unifies cloud and batch processing. It can be used for processing both streams of data and static files making it more relevant in the context of present day internet and social media. It is aimed at leveraging the present Hadoop platform and reducing the learning curve for development of applications over it. It is aimed at It can used through a simple API. It enables reuse of code by not having to make drastic changes to the applications by providing interoperability with existing technology stack. It leverages the existing Hadoop platform investments.

ACKNOWLEDGEMENTS

This paper has been written as part of a class assignment for the course: I524: Big Data Software and projects, Spring 2017, School of Informatics and computing, Indiana University, Bloomington. Special thanks to Professor Gregor von Laszewski, Dimitar Nikolov and all associate instructors for guiding through the process of writing this paper.

REFERENCES

[1] A. Kekre, "Apache apex blog incubator," Web Page, Sep. 2015, accessed: 2017-03-26. [Online]. Available: https://www.datatorrent.com/blog/apex-accepted-as-apache-incubator-project/

[2] Wikipedia, "Apache apex wiki," Web Page, accessed: 2017-03-26. [Online]. Available: https://en.wikipedia.org/wiki/Apache_Apex

[3] A. Kekre, "Apache apex blog introduction," Web Page, Sep. 2015, accessed: 2017-03-26. [Online]. Available: https://www.datatorrent.com/blog/introducing-apache-apex-incubating/

[4] Apache, "Apache apex application development documentation," Web Page, 2016, accessed: 2017-03-26. [Online]. Available: https://apex.apache.org/docs/apex/application_development/

[5] ——, "Apache apex application operator documentation," Web Page, 2016, accessed: 2017-03-26. [Online]. Available: https://apex.apache.org/docs/apex/operator_development/

[6] T. Weise, "Apache apex slideshare," Web Page, Jul. 2016, accessed: 2017-03-26. [Online]. Available: https://www.slideshare.net/ThomasWeise/apache-apex-stream-processing-architecture-and-applications

[7] Apache, "Apache apex," Web Page, 2016, accessed: 2017-03-26. [Online]. Available: https://apex.apache.org/

[8] S. Hall, "The newstack article on apache apex competition," Web Page, May 2016, accessed: 2017-03-26. [Online]. Available: https://thenewstack.io/apache-gets-another-real-time-stream-processing-framework-apex/

[9] Apache, "Apache spark," Web Page, 2016, accessed: 2017-03-26. [Online]. Available: http://spark.apache.org/

[10] S. Hall, "The newstack article on apache flink," Web Page, Apr. 2016, accessed: 2017-03-26. [Online]. Available: https://thenewstack.io/apache-flink-addresses-continuous-stream-processing/

[11] Apache, "Apache storm," Web Page, 2016, accessed: 2017-03-26. [Online]. Available: http://storm.apache.org/

[12] ——, "Apache samza," Web Page, 2016, accessed: 2017-03-26. [Online]. Available: http://samza.apache.org/