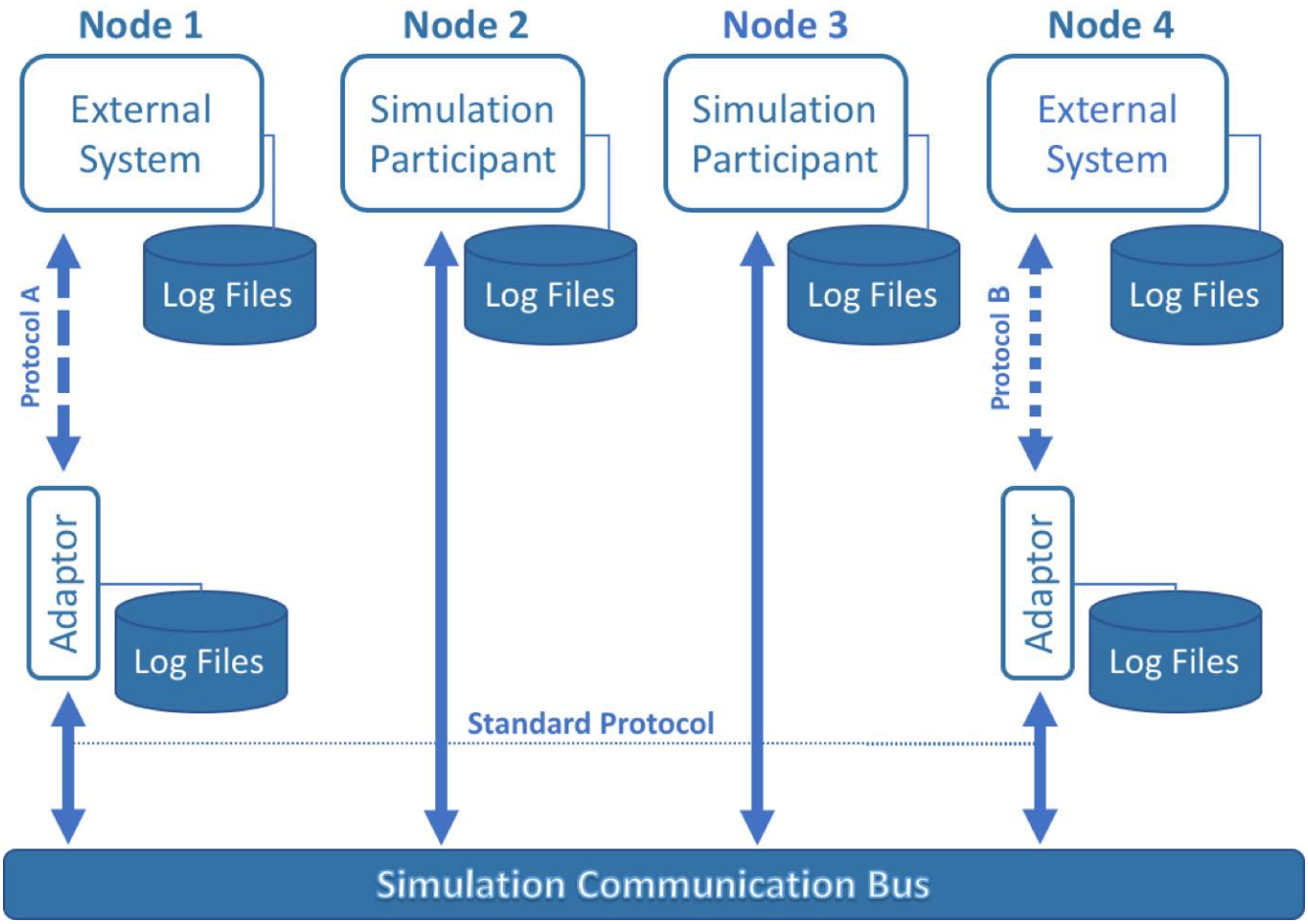
CohesionForce
TECHNOLOGY & SERVICES

A Modern Approach to Data Analysis

Through our years of progress in the development of large scale systems, we have experienced the pain of attempting to assess the behavior of these systems through the analysis of the data they produce. This paper provides an approach using modern open source technology to solve the problem of distributed data analysis.

To provide an example for this approach, let's assume that the large-scale system is a distributed simulation with several applications controlled through a common framework. All communication between the applications and framework are logged, as well as text logs and other binary traffic between the applications and external systems.



In this simplified example, each node represents a computing environment connected through a network connection and may be distributed across the US. The Simulation Participants represent applications like Simulation Control, Logging, Environment Modeling, Propagation, etc. The Simulation Adapters represent adaptations for existing external systems.

Also, there are several types of protocols that usually pop up in these systems. For this example, let's assume that Protocol A is DIS, Protocol B is defined by a set of CORBA messages, and Protocol C is defined by DDS topics.

Common Data Problems

Using the example shown previously, we can define the 4 major problems that we have repeatedly encountered when trying to analyze the data produced by these type of systems:

1. The Size of the Data – as the systems that we work with become more complex, the amount of data they produce grows. One system that we have analyzed generates almost 2 petabytes of data a year, and that amount is expected to increase.
2. The Location of the Data – the data that we need to analyze is often scattered amongst the machines used to run the system. In the example shown previously, the data is spread across 4 nodes. In the past, we have seen systems where there were more than 40 nodes involved in the simulation.
3. The Format/Schema of the Data – the format of the data varies between message types, application logging parameters, and other configuration settings. Each application included in the example system outputs a text log file indicating nominal or error cases. The data flowing into and out of the applications are also logged to binary files with the format dependent on the protocol used.
4. Data Duplication and Redundancy – Much of the data captured is a direct copy of the same item in another location or may be a slightly modified form of an item in another location. Again, referencing the example system, an event from Protocol A is transformed to the Simulation Protocol C, replicated on Nodes 2-4, and finally transformed to Protocol B going to the second external system. The same event is transformed twice and duplicated 3 times, further impacting the size of the data captured.
5. A 5th problem of data security is not addressed in this paper but must be accounted for in the configuration proposed for data analysis.

The Impact of These Problems

The increasing size of the data drives the constant need for **bigger/faster/expensive hardware**. The current approaches to data analysis rely on increasing the CPU speed, memory size, or specialized processor affinity techniques (based on operating system) and will not scale to accommodate the increasing size of data.

We have encountered some systems that have to pause occasionally to allow the data that has been produced to be collected and moved to another location for analysis. The distributed nature of the data forces a collection process to be applied to gather and move the data to a location where it can be analyzed. This will affect the schedule for running these large-scale systems, as the systems have to be stopped in order to collect the data. The overall impact is **decreased resource availability** and **increased lag** between the running system and the analysis of the data.

There are custom tools available for the different types of data that we use, but they are not extensible and **do not work together**. They also have problems when the **data format changes** due to updates in message formats or application logging.

Without some method of cataloging and indexing the data in a meaningful way, people are unable to identify items that can be culled. This results in large drives full of information that takes **more space than necessary** and **more time to search** than available.

CohesionForce Approach

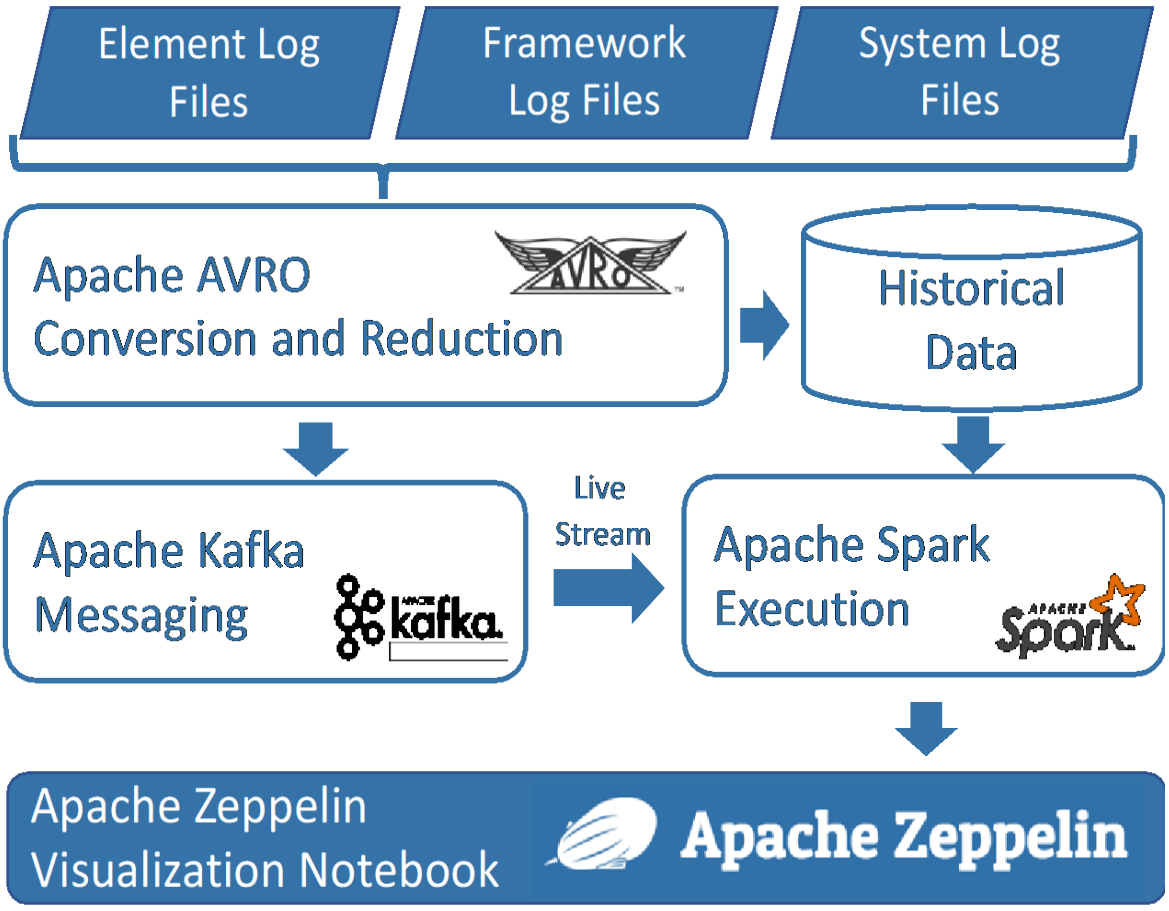
Through our work with open source projects provided by the Eclipse and Apache Foundations, we have created an approach that can be applied to resolve the problems that we encountered during our development and analysis of these large-scale systems.

We use model-based technology provided by the Eclipse Foundation to model the data schema and generate applications that can log and transform data. This allows us to mitigate the issue with data schema changes over time, as we can also generate applications to migrate legacy data.

To date, we have created loggers and transforms for DIS, HLA (FOM based), and DDS (topic based) protocols. We have also developed tools for these protocols to assist analysts with visualizing data relationships and creating queries.

We transform all data into a compressed binary Apache Avro format that also stores the schema with the data. This allows most modern data analysis frameworks to load and query the data with no further effort. As an example, a 1GB DIS binary log file (98% EntityStatePDU) will be reduced in size by 30-45% and can be used directly in the data analysis runtime.

For the data analysis runtime, we have created a configuration of several Apache Foundation projects based on the operational need. The basic architecture is shown in the diagram below:



101 Quality Circle • Suite 140 • Huntsville, Alabama 35806
256-562-0600
http://www.CohesionForce.com
Revolutionizing the way systems are Designed, Developed, and Deployed

Veteran Owned Small Business

The Apache Avro Conversion and Reduction is performed by our generated loggers and transforms. The resulting data can be stored locally, or in a shared data store.

The heavy lifting of the data analysis runs in the Apache Spark Execution cluster, which can scale based on the amount of data to be analyzed. At the top end, a configuration using this technology won the Daytona Graysort contest 2014 by sorting 100TB of data in 23 minutes with 206 nodes and 32 cores per node (http://sortbenchmark.org).

In its smallest footprint, this configuration can be deployed as a logger writing to a local file system, with a Spark Execution Worker embedded with the Zeppelin Visualization Notebook. We typically use this approach with a small (~20-50GB) data set to demonstrate the capability provided and allow analysts to build new queries and reports.

The Apache Zeppelin project provides a notebook-based approach for data analysis. Multiple analysts can work and collaborate at the same time, without the need of moving data to each analyst's workstation. The data can be queried through SQL (including sorting and joining) or through custom tasks written in Python, Java, and Scala. The list of available interpreters is growing with each release of Zeppelin.

The addition of Apache Kafka into the configuration provides a capability to perform live analysis of the data as it is being produced. Kafka is a messaging system that forwards data to the Spark cluster after it has been transformed to an Avro format. The same queries used for historical analysis can be executed as the data is captured providing a live view of the system.

As with Spark, the Kafka deployment can be as simple as a single process or scale to handle massive amounts of data with replication. For instance, according to the Netflix technical blog – "We currently operate 36 Kafka clusters consisting of 4,000+ broker instances for both Fronting Kafka and Consumer Kafka. More than 700 billion messages are ingested on an average day."

To manage the data security issues that may arise, other Apache projects such as Atlas, Yarn, and Hadoop can be added to provide access control to data and compute resources.

Data in a Perfect World

Legacy approaches to data analysis will only perpetuate the issues of more expensive hardware, decreased resource availability as data is transferred, and further dependency on custom proprietary tools that lock users into antiquated processes.

In contrast, a modern approach using the best available open source technologies will provide a scalable & open data analysis architecture with no license fees, no ties to vendor specific tools, distributed data access, collaborative notebooks, and responsiveness to data format changes.

As we have progressed, CohesionForce has been publishing information relating to the data analysis architecture and tools developed at various conferences. These include EclipseCon, Huntsville Big Data & Data Analysis Meetup, and AlaSim. If you would like more information or would like to schedule a technical interchange or demonstration, please contact us at solutions@cohesionforce.com.