Self-Describing Data Streams using a SOA approach for Collaborative Data Discovery

Scott Klasky, Hasan Abbasi, Manish Parashar, Norbert Podhorszki

In 2007, Jim Gray highlighted following concern, "Science faces a data deluge. How do we manage and analyze information?" He implied that science and our society faced a new grand challenge -- that of extracting knowledge from a flood of electronic information. He recommended the seamless integration of science and computing, and highlighted the need for better tools for data curation, data analysis, data handling and data visualization. Today, we see data from experiments and simulations almost doubling every year, much faster than Moore’s law, and one of the ways for scientist to handle this increase of data is to increase the number of scientist looking at the datasets that were once traditionally analyzed by a single scientist. Clearly the only way to make this sustainable is by creating tools and technologies which can facilitiate this collaborative nature of analysis, and that can scale with increasing data sizes, data complexities, and with the number of people collaborating. Furthermore, storages sizes are increasing at a much slower rate, and as a results, data will also need to be reduced in-transit to the extent possible. Thus, collaborative data analytics is quickly becoming an important area of scientific research due to the overwhelming size of data produced by both scientific instruments and simulations. One of the most important aspects of the collaborative effort is that most forms of data operations need to be cataloged and placed in the data streams/chunks independently from one-another. Data streams/chunks can come from individual diagnostics hooked up to a large experiment, from individual processors of a large simulation, or from smaller analytics routines that transform the data. Processing these data streams, we believe, must occur *in transit,* in order to avoid performance bottlenecks and provide energy savings by avoiding persistent storage through the file system. It is also critical to store the data-lineage in the individual data streams by capturing this information automatically as the data is generated. This automatic capturing is essential because the requirement on researchers to add metadata manually places an unrealistic burden on them. Also important to the platform is associating this metadata with individual streams, but providing the capability of aggregating streams into a multi-variable streams. Combined with semantic preserving efficient data movement techniques, these characteristics will allow for *in transit* operations that can greatly aid in sharing data and accelerating science.

This use case is different from techniques involving descriptive files such as HDF5 and netCDF, where the entire file has a single header containing metadata for the dataset. Quite often researchers can create separate metadata files which contain information in each file, thus building a data management system around these file formats As we move to larger datasets, we need to ultimately abstract away the notion of files from data streams, and be able to capture metadata *in-transit.* Here data streams can be *data-chunks*, where in a simulation groups of data from individual processes are chunked together in larger chunks, and common meta-data to each chunk can allow for efficient operation.

Enhancing the lineage, the data streams may also contain embedded workflows that provide the consumers of the data access to the actual operations performed on the data. Additionally the data stream can include hints, or non-binding *intentions*, on the best techniques for further processing the data. Embedding *history, operations* and *intention* alongside the data description enhances collaborative efforts by providing a more complete picture of the origins, lifetime and condition of the data set. Each data stream can contain different operations, which can be linked to other data streams or handled independently. Data streams can also contain embedded code or high-level description of the processing algorithm, which can be dynamically executed by the consumer, but delivered by the producer, encapsulating the concept of moving work to data.

Therefore, research in this area should include

* Methods to efficiently move data streams independently from one-another.
* Methods to capture metadata in each data streams according to the operations performed on the data streams.
* Methods to embed workflows and code to data streams and execute those *in-transit*
* Tools to efficiently write data streams to persistent storage.
* Methods to collect metadata in streams in a collaborative environment.