Chapter 1

The AOD format

General overview here, comparison between the different data types (Raw, CTF, ESDs, AOD) produced by the experiment and why they are needed.

1.1 AOD Production

How AODs are generated in the pipeline. Run3.

1.2 Analysis

Generated AODs will be stored and processed in dedicated *Analysis Facilities* or on local machines. These Analysis Facilities are specialized datacenters, part of the ALICE Grid (Note: Grid or GRID?). Physicists can submit different tasks to the grid to run against a large (or small) set of data. The Grid software automatically handles the scheduling of tasks and the IO of the AODs. Tasks may be split up over multiple machines or even multiple Grid *sites* at once, the smallest level of granularity the scheduler can handle is a single AOD file. Multiple tasks, by possibly multiple users, are scheduled together in a so-called *Analysis Train*, this groups (sub-)tasks together based on the accessed files thereby reducing file IO and data movement.

Each sub-task runs as it's own separate process. The scheduling of sub-tasks is fully automated and transparent to the task. Tasks can communicate between each other and other actors using a messaging system build on top of FairMQ(CITE) (which is in build on top of ZeroMQ and nanomsg(TODO: check, CITE)). This can be leveraged to share a single in-memory AOD between several consumers.

1.3 Requirements

Before the start of the project a list of all necessary features was compiled, these included:

- **Pruning:** The ability to reduce an AOD to a smaller subset by removal of structures and/or components of structures. This is needed in order to allow the physicists to reduce a larger dataset to a much smaller one for subsequent (local) analysis and reduce the required IO- and memory-bandwidth at runtime. For example, the most commonly used object stored in an AOD file is a detector *track*. This is the reconstructed trail left by a particle in the detector. A single track contains many different types of information such as its point-of-origin, it's momentum, the covariance matrix of the reconstruction, references to other associated objects etc. In most cases however, one is not interested in all of these components. Sometimes one might only be interested in a single component of the momentum. Pruning is the act of getting rid of all other data and, in this case, leaving only a single component of the momentum.

- **Skimming:** The ability to infer certain facts about an AOD without having to touch all the data contained therein. For example one might only be interested in tracks which have a positive charge associated with them. Skimming would be the act of reading/copying only those tracks which fit this criteria. (TODO: write psuedo-code for these to use as an example, i.e. $if(charge <= 0)\{continue;\}$)

- **Growing:** The ability to easily extend an AOD with additional data, structures or components. For example, one might like to extend the track structure with a new field to cache the result of a long and complex computation, call it *'HiggsBosonCandidate'*, that would be growing the track structure. Another option would be the adding of a whole new struct to an existing AOD, for example one might have a set of simulated (Monte-Carlo generated) collisions and wish to extend the AOD file to include simulated tracks. (TODO: write better good)

This list was later extended to include:

- **Derived Types:** Certain components are rarely used and/or take up a large amount of disk-space when stored and can be recomputed from other stored data if needed. It is therefore favorable if these could be recomputed transparently on a as-needed basis as opposed to being stored in-file.

- **Zero-copy message parsing:** Previous work(CITE,Mikolaj?) has shown that a large performance penalty is incurred by data-serialization. In order to avoid this overhead the format should be able to pass data around without any (unecessary[1]) serialization

[1] One cannot avoid (de)serialization when passing data between machines with different

- **Maintain a similar interface:** The current ALICE software framework is used by X(CITE) users, contains Y(CITE) lines of code, and was written over the course of Z(CITE) years. While it has been deemed possible, even neccesary, to convert parts of the framework to work with the new format a dramatic change in the way physicists interface with the data would involve to much man-hours in terms of rewritting existing software and utilities and forcing every user to adapt to the new framework.
- **Schema Evolution:** On occasion, changes might be made to the way in which data is stored. This might be the conversion from a single float to a 3-dimensional vector or it might be a change in coordinate system to improve accuracy and/or compression. Changes of this kind are called Schema Evolution, and it is important that the framework knows how to handle this so that it can adapt and read both old- and new-formatted data.

native data types, such as a big-endian and little-endian machine but that is out-of-scope for the AOD.