Event Mixing and Online Reconstruction in the PANDA Detector Simulation

Sebastian Neubert, Roland Kuhn
Technische Universität München
Physik-Department E18

June 3, 2005

Abstract

This note describes the concept of a data stream based processing chain in the PANDA simulation and reconstruction environment. Such a system is needed for the online reconstruction software of the PANDA DAQ and can be used to facilitate studies of the effects of event mixing and pileup in the PANDA detector system. These topics are of special concern to a high rate TPC as central tracker option but may be important for other detectors as well.

1 Event Mixing in PANDA

At the design luminosity of PANDA ($\mathcal{L} = 2 \cdot 10^{32}\,\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$) the interaction rate will be in the order of $10^7\,\mathrm{s}^{-1}$. The average time interval between two events is in the order of 100 ns. Although the time resolution of many detectors is of course better than this, the finite time windows (event life time) of most detector systems will cause at least some events to overlap. The event life time in a detector is the time interval in which the detector produces signals which are correlated to one primary interaction in the detector system. Usually this is a very short time. Particles that are produced in primary interactions traverse the detector system nearly at the speed of light. The time span it takes all particles to leave the detector (or to decay or be absorbed) is in the order of some ten nanoseconds. However depending on which technology is used for the detection of the particles some time will pass until the detector has produced a signal. Which timescales play a role here depends on the involved detector specific processes: drifting charges, long electronical integration times, etc.
An extreme case is the time projection chamber. Due to the long drift paths (max 1.5 m) considerable time will pass after a charged particle has entered the TPC until a signal is actually created. One drift cycle is the maximum time an electron needs to reach the readout plane, i. e. the drift time over the complete 1.5 m which is $\sim 50\,\mu$s depending on the gas mixture that is used. In order to be sure to get all information from one event out of the TPC one has to wait $50\,\mu$s. But also for a normal drift chamber with typical dimensions in the order of one cm, the event life time can reach values as large as a few hundred nanoseconds.
As long as the rate of primary interactions in the detector system is such that in most of the cases the time between two events is larger than the event life time in the slowest subdetector there is no problem with event mixing. All events can be separated in time without problem. However at high rates this is not true anymore.
For PANDA with an average time between two primary interactions of 100 ns there will be few subdetectors where the event life time is shorter than this. (Again it has to be emphasized, that event life time must not be mistaken for time resolution!)

2 The Event Paradigm

For the further discussion it will be useful to have a closer look at the meaning of the word "event" (in the context of a particle physics experiment). There are several different entities which are designated by this term:

- The subject of our studies are interactions of subatomic particles. The classical scattering experiment is viewed as a momentary encounter of the projectile with a target particle. In this collision the particles are deflected or even destroyed and new particles might be created. The products of this process travel through the detector system. In this context the term event denotes one single projectile-target encounter and the collectivity of the scattered particles.

- In the analysis of such experiments one tries to learn about the underlying processes by looking at the scattering products that have been registered in the detector system. The observables are the momentum 4-vector of each particle and the space-points in the laboratory where individual particles have been created or where they have decayed – usually called vertices. In this context an event denotes the collectivity of 4-momenta and vertices that are *believed* to correspond to one scattering process. However it is a long way to get to this point in the analysis.

- In the past, hardware trigger systems have been used to determine the *time* when a signal (or certain patterns of signals) has been created in the detector signifying the traversal of particles through the experiment. In this context an event is simply a notification that something (interesting) has happened in the detector at a certain time. All data which is produced in a time window near (maybe with a constant offset) the trigger notification is marked to belong to this triggered event. At high rates this time window may get so big compared to the time between the primary interactions that a lot of events fall into it. In principle one could define overlapping time windows, each corresponding to one event candidate. But this would multiplicate the data, because data packages would be assigned to a whole set of events.

- At the reconstruction stage there is even another interpretation of the term event. Here the task is to group individual signals from different subdetectors into collections (tracks) which have been created by one single particle (at least to our best knowledge). An event is then usually defined by collecting all reconstructed tracks together for which the same point in time of creation has been determined. Traditionally this is given by the trigger and so the grouping into events is already done on the raw-data level.

One thing that is common to these definitions is that an event is defined at a certain time. However we observe, that this connection to its time is only important to define the event during the data taking. For the analysis of the physical process the time when an event has happened does not play any role. One might argue, that these distinctions are purely technical. Indeed, if primary interactions are well separated in time the term might be used for all the points mentioned without any problems. However this changes if event overlap is present at high rates. The weak point is the definition of the trigger event. If the rates are so high, that in any time interval, which contains all signals from one specific primary interaction there are also signals created by particles created in other projectile-target encounters, then the trigger event cannot be defined like above. In this case more elaborate methods are needed to analyze the data and to define events. This is the reason why the PANDA DAQ will not define a hardware trigger. The raw data is not divided into chunks which represent one triggered event. Instead continuous streams of data are handled.

3 Dropping the Event Paradigm: Data Streams

Maybe the most basic way of looking at the data that is delivered by a detector system is that of a data stream. Without knowing what is going on in the machine, without a priori imposing the concept of individual events one can simply speak of the data one gets from the experiment as a (more or less) continuous flow of information. (There might be actually a couple of independent streams, corresponding to the individual subdetector systems.) This data stream consists of a series of data packages containing the digitized signals from the detector subsystems. The distinct property of this data stream is that the packages appear according to their time of creation.

Of course we believe that there have been individual primary interactions in the target, which we want to extract and analyze. So we look for correlations in the data stream. An event is therefore defined by the correlations it creates in the data. Most important here is of course the time correlation. However this time correlation need not necessarily be realized on the level of the raw data packages. In general one has to preprocess the raw signals before actual correlations become visible. An example is the TPC, where on the level of the raw data there is almost no correlation between signal time and corresponding primary interaction. It is important to realize, that it is the high rate compared to the finite life time of events in the detector system, which destroys the time correlation on the raw data level.

This means, that one has to use additional information (additional correlations in the data stream) to define events for the following analysis. The most important point in this consideration is that *an event will now be defined in the reconstruction process. It is not self-evident, if the so-defined events do have any connection to the elementary processes which we want to study. Instead it is the task of the reconstruction algorithms to ensure this connection.*

This is what is called the event deconvolution. Again: One should not make the mistake to believe that a reconstructed event automatically corresponds to a primary interaction in the experiment. This distinction should also be reflected by the structures which are used to organize the data.

Because the raw data rate is much too high to be recorded onto tape one wants to have to possibility to sort interesting events from background online during the data taking. This is the primary objective of the trigger. Of course to do this an event has to be defined first which can be either discarded if it is identified as background or written to tape. So the event deconvolution has to be performed as part of the DAQ in an online reconstruction system. The trigger logic is then implemented completely in software modules on top of the online reconstruction.

The software trigger approach has further advantages besides the ability to cope with high rates. Since all trigger algorithms are realized in software and are based on (at least partially) reconstructed events a high degree of flexibility and versatility can be achieved.

Obviously the methods used to realize such an online reconstruction and software trigger, which allows to run at extreme rates are of prime importance to the success and performance of the experiment. Therefore the rest of this note is presenting some ideas how the concept can be included into the PANDA software design at an early state to allow development and verification in a unified framework.

In order to systematically study how event mixing will affect the experiment the PANDA simulation must include these aspects. A unified approach is mandatory to avoid inconsistencies between simulation and real data. This means that the software to process simulated data must be the same as what is used on real data – also on the online processing stage[1].

4 Aims of the Simulation

At first let's try to define some aims for the simulation and derive requirements, which the software has to fulfill to reach these goals.

A computer simulation should give answers to the following questions, concerning the event mixing problem:

1. Which computational technologies can be used to do event deconvolution? The key challenge of the continuous readout is the ability to build events from the data stream.
2. Which efficiencies can be reached for the correct assignment of data to individual events?
3. How pure can the assignment be made (i. e. how many events will have false tracks assigned to them)?
4. How will the detector design — especially the interplay between different detector components — influence the event building efficiencies?
5. What are the main parameters of the data quality that influence the performance of event deconvolution?
6. Which computational resources are needed to perform this in real time?

We think it is best to model the real situation in PANDA as closely as possible. This has the huge advantage that software developed for simulation studies can be reused for the processing of the real data. In fact simulation studies can only make reliable statements about the performance of the detector system if these crucial features are included. The same of course holds for the offline reconstruction which also will be used for simulated data as well as real data from the experiment.

Consequently the software has to provide the following functionality:

- Each generated monte carlo event can be assigned an event time. This can be done on the basis of an event rate which is given as a parameter of the simulation. The event rate is only important for the event mixing, it is not needed in the GEANT simulation.
- Each digit has to be supplied with a time stamp (created as part of the digitization process). This simulates the time distribution system which is planned for the PANDA DAQ.
- The data can be represented in a time ordered manner which allows to mix an arbitrary number of events together. In particular there has to be a dedicated module, which convolutes a series of simulated events into a time ordered data stream.
- There should be the possibility to make such data structures persistent.
- The monte carlo truth has to be kept in an event based data structure which can be used for efficiency tests.
- There has to be a dedicated part in the processing chain which implements the event deconvolution algorithms (since in the experiment this has to be done online we call this the online reconstruction). These modules have to work on serialized data without any knowledge of the simulated event structure.
- The online reconstruction software should allow the development of algorithms which can be used later in the software trigger.
- There has to be the possibility to define reconstructed events and to create associations to monte carlo events.
- The software must be able to digest streamed data as well as classical event based data.

[1] The online processing might also use hardware implemented components. These have to be modeled and emulated in software modules for simulation studies.

- It would be useful to have the possibility to benchmark the resource demand in the online reconstruction to be able to compare to DAQ implementations.

5 Upgrading the Framework: Data Streaming

The PANDA software framework (inherited from BaBar) offers excellent support of event based analysis. However data streaming is not foreseen at the moment. The software has to be expanded such that the online reconstruction can be developed in a unified environment. This section describes one possible ansatz for this.

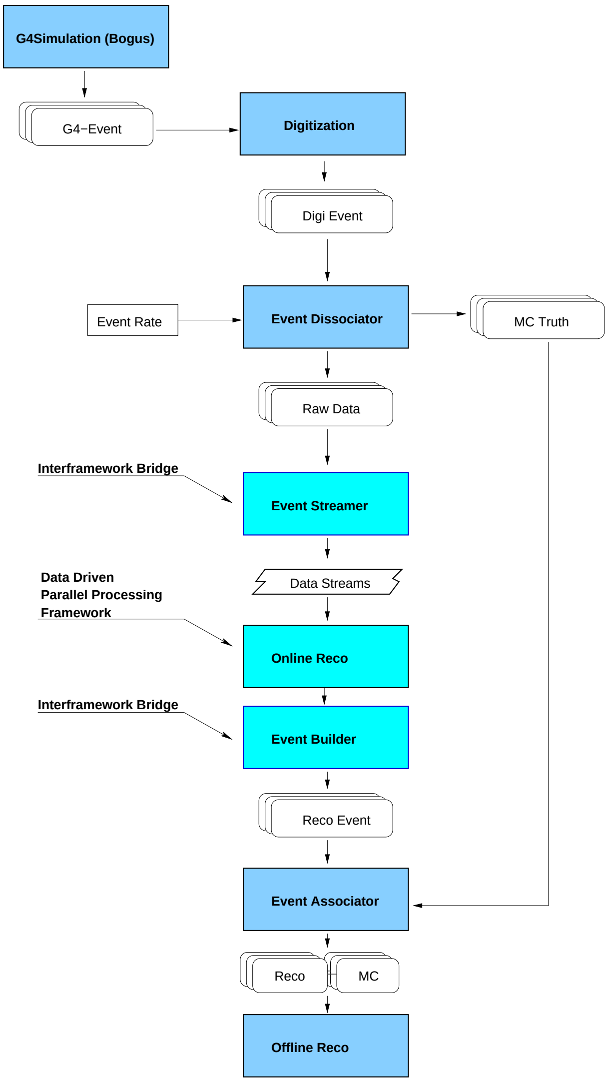

Figure 1: Dataflow in the PANDA software framework

The basic idea is as follows (refer to Figure 1): The simulation is done on an event by event basis including the digitization. One feature that has to be supplied at this level is a correct timing. Every raw digit has to be supplied with a generic time stamp. The simulation of a finite event life time in a particular subdetector is part of the digitization which for example takes into account drift times and integration times.

Before the simulated events can be serialized an important issue is the treatment of the monte carlo truth. This information should remain organized by events. It has to be dissociated from the raw data before the streaming. However the monte carlo truth events have to be supplied with the valid event time, so the event timing has to be done before the dissociation. In addition it might be useful to give each digit a monte carlo identifier, which points back to where that particular hit came from. Later, after online reconstruction and event building has taken place, the reconstructed events have to be associated again to the monte carlo truth. That means, for each reconstructed event one has to find the corresponding monte carlo event (based on the reconstructed interaction time and maybe monte carlo IDs in the raw data). Due to inefficiencies and impurities in the event building this association has to allow for missing tracks and ghost data in the reconstructed events.

After the separation of the monte carlo truth the raw data events are digested by a new framework component which for the moment will get the working title `EventStreamer`. This component is responsible for the convolution of the simulated events into a data stream. In order to achieve this it performs a buffering and time ordering of all the raw data packages. To reflect the structure of the PANDA DAQ it will be practical to create at least one data stream for each subdetector.

The data streams are processed in a dedicated software system. Here the complete event deconvolution process has to be implemented. This system also serves as the basic development platform for the online software. Section 6 goes into a little more detail about how data streams can be handled and what options are available for PANDA to realize such a system.

The final step in the online reconstruction is the event building. At this stage the data is already reconstructed to such a degree that tracks can be grouped into events. The `EventBuilder` interfaces to the event based structure of the BaBar framework again.

What is remaining is to create associations of reconstructed events to monte carlo truth as described above. After that the framework in its present form can be used to implement the offline reconstruction and the analysis tools.

While the purpose of the online reconstruction is focused on the event building, the offline reconstruction will try to achieve a high precision in the observables. Here the data can be reprocessed with sofisticated refitting and vertexing algorithms.

In our opinion this procedure is the best way to model the situation in PANDA. The event paradigm is dropped at that point, where it is actually violated. It is restored again where it is necessary to do physics analysis. In this way the components developed for the streaming part of the processing chain can be directly used as a basis for the future development of the online software.

6 Processing Data Streams

This section explains some basic ideas how to handle streams of data and proposes one solution which could be used to add streaming support to the BaBar framework.

6.1 Principle of Data Driven Processing

The output of the `EventStreamer` is N data streams corresponding to N subdetector systems[2]. Dedicated modules process the streamed data, each module performing one specific task, quite similar as in the BaBar framework. The handling of streamed data however is different from handling events. Individual data packages become much smaller down to the single digit level. The order in which data is processed is not defined so clearly — several data streams containing data packages from several events might even be processed in parallel. It is no longer possible to call a function like `ProcessEvent` for each module in turn.

Instead a data driven approach to handle the situation seems appropriate. This means, that computing modules try to process the data as soon as it is available and hand it down the processing chain as soon as they are finished. Figure 2 illustrates this principle. The individual modules are running as independent processes (or threads) on their host computer. As soon as a module is supplied with data into its input buffer it tries to digest the newly available information according to its processing algorithms. A module might for example receive tracker hits and build track candidates from those. Every time a new hit is available, it is added to a compatible track candidate (or a new candidate is created). When a track candidate is finished (e.g. when a kink is encountered or the detector boundary is reached) it is placed in the output buffer and made available to the following processing modules. Each module can have access to an arbitrary number of data streams and it can produce as much output streams as necessary. A module can either modify the input data stream or it can even create completely new data structures and put them into a new stream (like in the track finder example above).

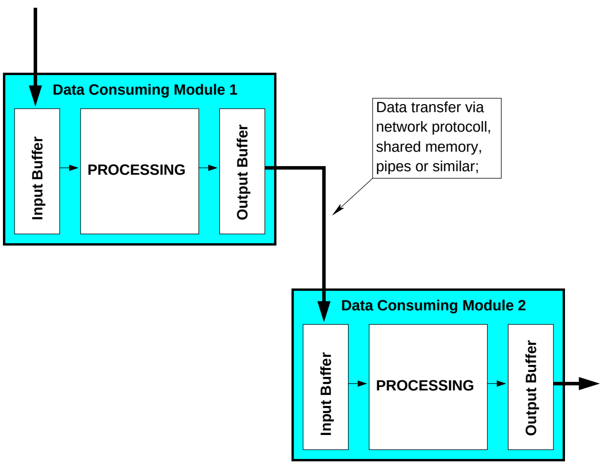

Figure 2: Data driven architecture

[2] In fact each subdetector system could be further subdivided leading to even more separate data streams

6.2 Providing Streamed Data Handling Services: The ALICE HLT Data Transport Framework

A framework that supports the buildup of an data driven processing architecture has been developed for the ALICE High Level Trigger (HLT) which faces similar difficulties in terms of data rate as PANDA will. The architecture and the available tools are described in [Ste]. This software bundle is freely available and could be used as a basis for the extension of the PANDA framework to support data stream processing. The author actually mentions PANDA explicitly in his work as an experiment that could make use of these tools.

The main advantage of such a system is that it supports the parallel processing of data streams which can be defined in an almost completely generic way. It is also easily scalable to multi node processing farms which are necessary to deal with high data rates.

The ALICE HLT data transport tools use three different mechanisms for the communication between different processing modules. Communication across a network of computing nodes uses the TCP protocol while for the data interchange on a single node shared memory and named pipes are used. Several templates for the creation of complex processing topologies are already available, such as data sources, processing modules and data sinks (modules which write data to disk). Another aspect (being of big importance if the system is actually used online) which is also supported is fault tolerance. For example it is foreseen, that the processing network reorganizes itself should one of its (redundant) nodes fail.

Figure 3 shows a simplified data flow diagram of a processing network for the PANDA online reconstruction that could be realized with the ALICE HLT data transport tools. The picture is intended to

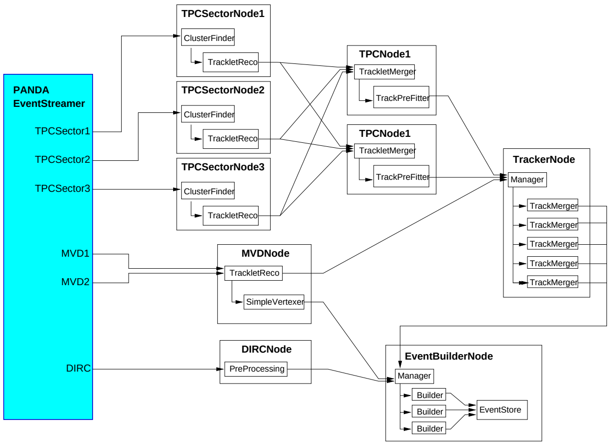

Figure 3: Sample topology of PANDA online reconstruction

give an idea how parallel processing of data streams could be organized. TPC, MVD and DIRC data is processed in parallel on a couple of dedicated computing nodes before their data is collected on an event builder node. The reconstruction process is split into several distinct modules which are linked by data transfer lines.
Only three subdetector systems are included here. In reality of course all subdetectors have to be processed by the online reconstruction in order to be able to do event building.
The details of the processing network topology have to be optimized for the PANDA requirements. However this is out of the scope of this note and remains for future investigations. The primary feature of a system like that we are interested in is the effective processing of streamed data.
For simulation purposes all module processes can in principle run on the same computer. But it is important to note that the online reconstruction will be an expensive operation in terms of CPU power and memory. Therefore it might be desirable to implement a parallel processing network distributed on a number of nodes already for simulation studies.

6.3 Interface between Data Handling Frameworks

The software framework from BaBar provides services to construct applications for event based data processing. Its key elements are the application framework and the event store. The application framework allows to build processing sequences from individual application modules which can be configured using TCL scripts. The event store allows to efficiently organize transient as well as persistent data in an event based structure. This framework will be used to create the simulation, the digitization and the offline software for PANDA.
The ALICE HLT data transport framework provides services for the creation of a modular parallel processing system for serialized data. It is a promising option for the development of the PANDA online reconstruction software.
There are two clear interfaces between those two frameworks. In Figure 1 the corresponding modules have been marked as inter framework bridges. The `EventStreamer` converts from event based data to streams while the `EventBuilder` creates events which can be handed back to the BaBar event store. The bridges are part of both frameworks and share their services.

7 Conclusion

We have discussed the importance of event mixing which will be a real challenge for PANDA if one wants to process the high interaction rates that are expected from the experiment. At high event rates the finite time windows of the subdetectors will lead to an overlap of data from different primary interactions. To handle the situation the event paradigm has to be dropped in the data acquisition chain. The PANDA DAQ is based on a "triggerless" hardware architecture. Instead the concept of data streams is introduced. Every subdetector delivers a stream of data packages to the DAQ system, which are marked with precise time stamps. This data is processed by the online reconstruction in a massively parallel manner to cope with the resulting rates. Events that can be used for physics analysis are defined during the reconstruction (and not by a hardware trigger) which has to be performed online in order to be able to apply a software trigger/filter for data reduction.
All these concepts are critical for the performance of the whole experiment and their feasibility is a prerequisite for the operation of a continuously read out high rate TPC. It is therefore necessary to develop methods and algorithms which tackle those problems already at an early stage in the software development process.
We propose to include a data stream processing chain into the PANDA software environment. One possible starting point for the development could be the ALICE High Level Trigger data transport framework.

References

[Ste] Timm M. Steinbeck. A Modular and Fault-Tolerant Data Transport Framework. *Computing Research Repository (CoRR)* **cs.DC/0404014** . http://arxiv.org/abs/cs.DC/0404014.