ALICE DAQ Internal Note

02 April 2002

Pierre.Vande.Vyvre@cern.ch

Summary of first meeting on rare triggers (02 April 2002)

Present: T. Anticic, C. Fabjan, L. Musa, K. Safarik, J. Schukraft, P.Vande Vyvre

---

1. Introduction

1.1. Do we really have a problem with rare triggers?

The result of the first batch of simulation done by T. Anticic has shown potential problems with the way rare triggers are handled. The problem arises inevitably whenever the data volume sent over the DDL's (as given by the trigger mix and event size) exceeds the data volume that can be processed by the DAQ/HLT system. In this case, the buffers in the intermediate stages of the DAQ (GDC's, LDC's, or RORC's) get full and by the action of the back-pressure this is propagated up to the detectors that become busy almost all the time. As soon as the busy is released, the frequent triggers are obviously the most likely to fire and to put the detector into busy state again.

This is just a comment and it does not in any way question the validity of the conclusions made in the note.

The rare event problem is *also* caused by the lack of multi-event buffering in the sub-detector front-end electronics. Any cluster is as bad as the worst sub-detector it contains and there is a number of sub-detectors which can only handle one event at a time and require a substantial time to recover from a negative L1 decision. The simulation might not show the effect because it uses an "idealised" model of the sub-detector response and can hardly be blamed for it since a detailed description of the ALICE sub-detector front-end electronics *has not yet been compiled.*

Fortunately, the proposed scheme deals with both causes: the DAQ/HLT throughput and the inefficiency of sub-detector front-end electronics.

1.2. Requirements for a solution

Two requirements must be met to solve the problem:

- Keep the detector lifetime as high as possible in order to catch as many rare triggers as possible;
- Keep the backend part of the DAQ system (event building and mass storage) as busy as possible, i.e. as close as possible to the maximum installed bandwidth.

2. Implementation

A note circulated by Pedja and proposing a feedback mechanism from the DAQ to the CTP has been used as input for the discussion.

2.1.1. Downscaling

The downscaling currently foreseen in the CTP will allow limiting the rate of each trigger class. It would be possible to adjust the downscaling so that the adequate rate of frequent triggers is obtained.

> *"Balancing"* the rate of trigger classes is probably more adequate than *"limiting"* - another pedantic nonsense...
>
> I will now abuse the attention of the "Rare-event Think-tank" and make a digression to re-visit the general mechanism of downscaling. *Is it, as it is, good enough?*
>
> A brief reminder of what we have now - text taken from the version 1.0 of the **CTP URD** (not yet fully updated):
>
> **1.16.1** The CTP shall include individually programmable pre-scalers for each **L0** class-trigger. These will pass every **N**th occurrence of the class-trigger, where **N** is a programmable number.
>
> The maximum value of the pre-scaling factor **N** shall be **1,048,575** (20 bits).
>
> **1.6.1** The CTP shall generate internally **2** random-trigger signals with independently programmable rates in the range of **0.1-100 kHz**. The signals can be combined with other trigger conditions, or used inclusively.
>
> **1.6.2** The CTP shall generate internally **2** pre-scaled **BC** clock signals with independently programmable pre-scale factor in the range of **1-16·10^6** (24 bits). The signals can be combined with other trigger conditions, or used inclusively.
>
> Note 1: The pre-scale factor must not be a multiple or a sub-multiple of the number of bunches in an LHC orbit (3564) if one wants to sample all bunches equally.
>
> The 2 random triggers and the 2 pre-scaled **BC** clocks are a global facility, shared at any time by all the 50 trigger classes.
>
> Once again: is it sufficient? (I remember Jürgen being worried about the *resolution* of the adjustments.)
>
> *Comments please!*

There are however two main drawbacks:

- During a fill of the LHC, the luminosity will decrease by a factor of between 2 and 4) and it will therefore be required to adjust the downscaling factors during the fill.
- It will be needed to know the history of the changes of the downscaling factors for the normalisation during reconstruction and analysis.

> From the LEP experience - similar problems were usually resolved by breaking a fill into a number of runs.

2.1.2. Feedback mechanism from DAQ to CTP

The solution consists of making a time-sharing of the detector lifetime between periods dedicated to frequent triggers and rare triggers and periods reserved to rare triggers only.

The solution is based on the Class Mask bits in the CTP. Each trigger class has a corresponding Class Mask bit that must be enabled for the corresponding class to fire. Two thresholds would be defined for the LDC's buffers: a high-level threshold or almost-full and a low-level threshold corresponding to almost-empty.

When the almost-full threshold is reached, the Class Mask bits corresponding to frequent triggers are disabled and the system is reserved for rare triggers. Given the

total amount of buffering in the system, the DAQ back-end will be kept busy for a while.
When the almost-empty threshold is reached, the Class Mask bits corresponding to frequent triggers are enabled and the system is open to all triggers. The occupancy of DAQ buffers will increase again.
Data-taking periods will then be time-shared between two types of behaviour. When all types of trigger are enabled, the LDC's buffer occupancy will increase rapidly until reaches the almost-full threshold is reached. This will then be followed by a period reserved to rare triggers. The LDC's buffer occupancy will then decrease because the rare triggers do not saturate the DAQ bandwidth.
One can estimate that the PC's available at startup time will include several Gbytes of memory. If one assumes that 1 Gbyte is dedicated to the data buffering, one TPC LDC is able buffer of the order of 3000 events (70 MBytes collected by 216 DDL's). Even at the maximum acquisition rate of 200 Hz, the time constant is of the order of 15 seconds. In these conditions, the solution proposed would not require a separate low latency network to carry the feedback messages.
A big advantage of the proposed solution is also that it is based on existing hardware in the CTP.

This is now about the implementation and, in order to make sense, it has to be detailed; it mainly concerns Pierre (DAQ/HLT).

I'm making the following assumption: there is *only one "flavour"* of rare events and there is no need to *selectively and independently* "boost" the acquisition of "rare", "rare-rare", "golden", or ... rare event groups. Sounds logical, but *please check and confirm.*

In that case, I would prefer to leave the Class Mask bits alone and introduce another signal - a mode selector bit *All/Rare* (**A/R**). In the *All* state - all the classes are enabled; in the *Rare* state - only the classes associated with the rare events are active. The signal would be an additional ***L0*** *trigger veto,* common to all the classes - see **Figure 1**. (Another digression: class pre-scalers position is now fixed *before vetoes* since, according to Orlando, the trigger bias, dependent upon the location of pre-scalers in case of ATLAS, does not exist in ALICE. *Everybody agrees*?). At the CTP, the signal has to be synchronised with the **BC** clock (a personal reminder).

The **A/R** signal shall be generated and controlled by the DAQ/HLT system. It could be done in a number of different ways, all *equally successful* in maintaining the DAQ/HLT event processing throughput at its top level and allocating the maximum time to the acquisition of rare events (see **1.2 Requirement for a solution**). The choice of option shall eventually be made by the DAQ/HLT group(s), but the decision is (moderately) urgent since, in some cases, additional CTP hardware might be required. In order to speed it up, I'll list some possible implementations.

First, I'll indicate the assumptions made about the rates - *please check and comment.* The *rate of rare events* ($\mathbf{f_R}$) is significantly lower than the *rate of all events* ($\mathbf{f_A}$), dominated by balanced (scaled down) rates of "normal" classes. The *processing rate* ($\mathbf{f_P}$) of the DAQ/HLT system is sufficiently lower than the $\mathbf{f_A}$, which makes the proposed scheme possible (the $\mathbf{f_P}$ higher than $\mathbf{f_A}$ doesn't make sense - even if we had enough money to do it).

Figure 2 shows the behaviour of the system controlled by a simple (simplest) servo that monitors the occupancy of the LDCs and keeps it within the arbitrarily set limits *Nearly Full* and *Nearly Empty*; the shape of the **A/R** signal is also presented. This is the scheme described in this section (**2.1.2**) and in my "proposal" (***Principle***).

Another approach is to apply a *dynamically optimised time function* - brief description is given in the "proposal", section ***Another comment on implementation***. The function is generated by selecting a convenient *period* and continuously optimising the *duty cycle* in order to keep the LDC occupancy around the 50% mark. The "ideal" behaviour and the corresponding shape of the **A/R** signal are shown in **Figure 3(a).** In reality, the occupancy will drift up and down, as a result of rate variations and the corrective actions - **Figure 3(b)** (the "see-saw" variation of the occupancy and the corresponding shape of the A/R signal are represented as wide red/grey lines).

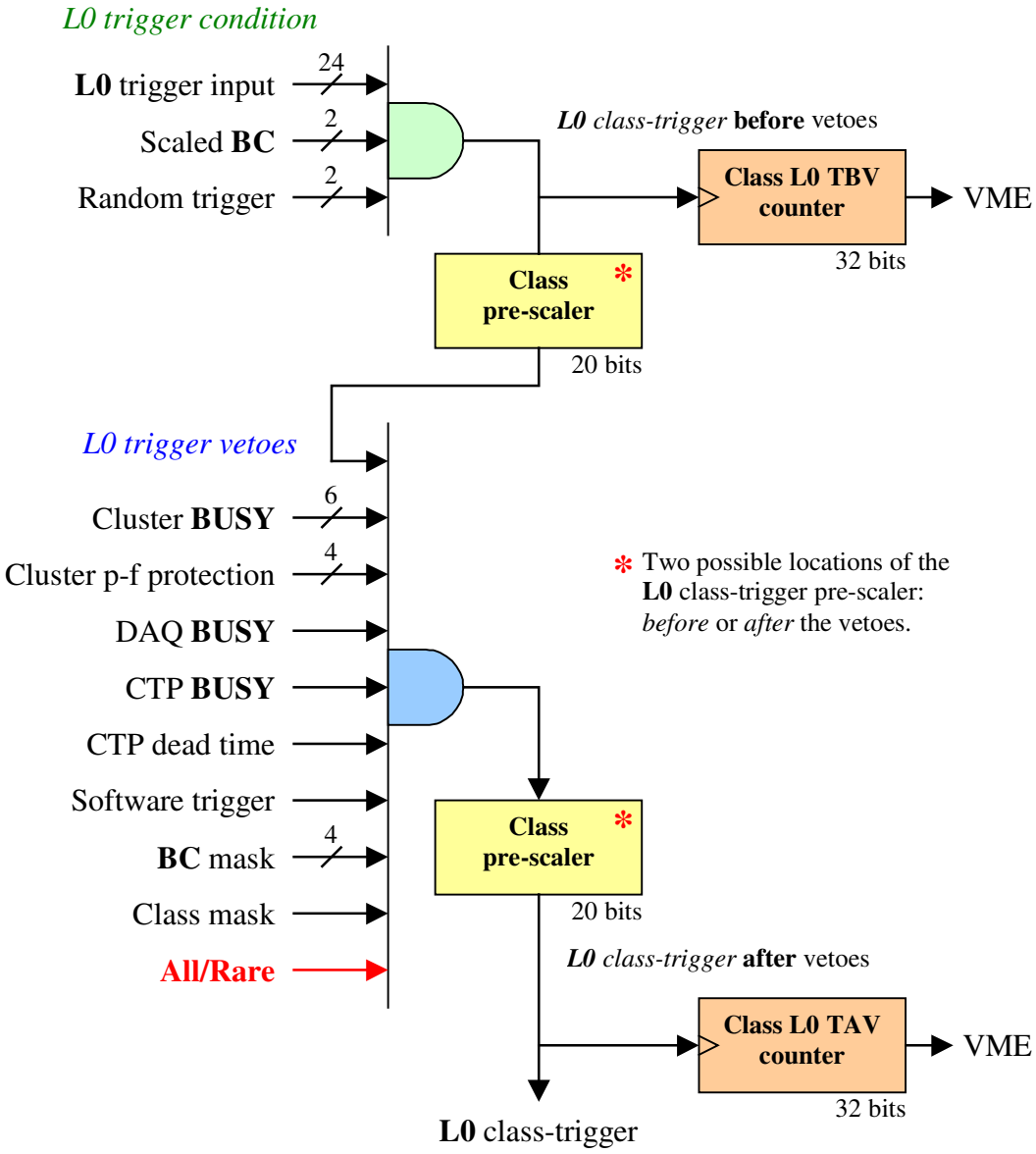

Figure 1 **L0** class-trigger logic

The servo with optimised parameters is likely to keep the LDC occupancy under control for a length of time without any intervention, but the behaviour of the system must be continuously monitored and the corrective actions would be required from time to time. The DAQ/HLT system could *continuously* (frequently enough) monitor the occupancy of the LDCs and undertake a corrective action (*Up-kick*, *Down-kick*) only when the preset limits (*Nearly Empty, Nearly Full*) are exceeded - **Figure 4**; the history of required corrections could indicate that the re-optimisation (change of the *duty cycle*) is needed.

Alternatively, the servo could sample the occupancy in regular intervals (**Figure 5**) and for each sample perform an appropriate proportional correction; with a high capacity of the LDC buffers, the stability of the sampling interval is not critical. Again, the history of corrections could trigger the re-optimisation.

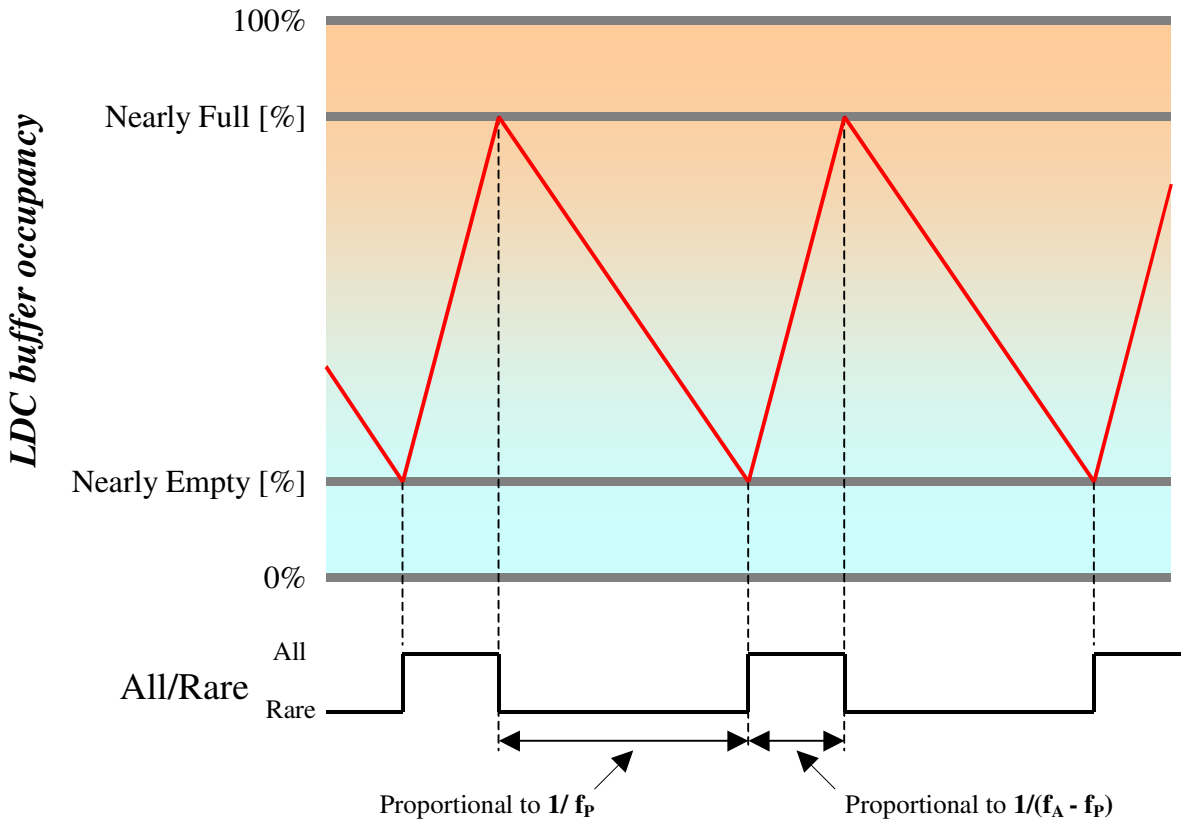

Figure 2 The All/Rare signal generated by a "bang-bang" servo

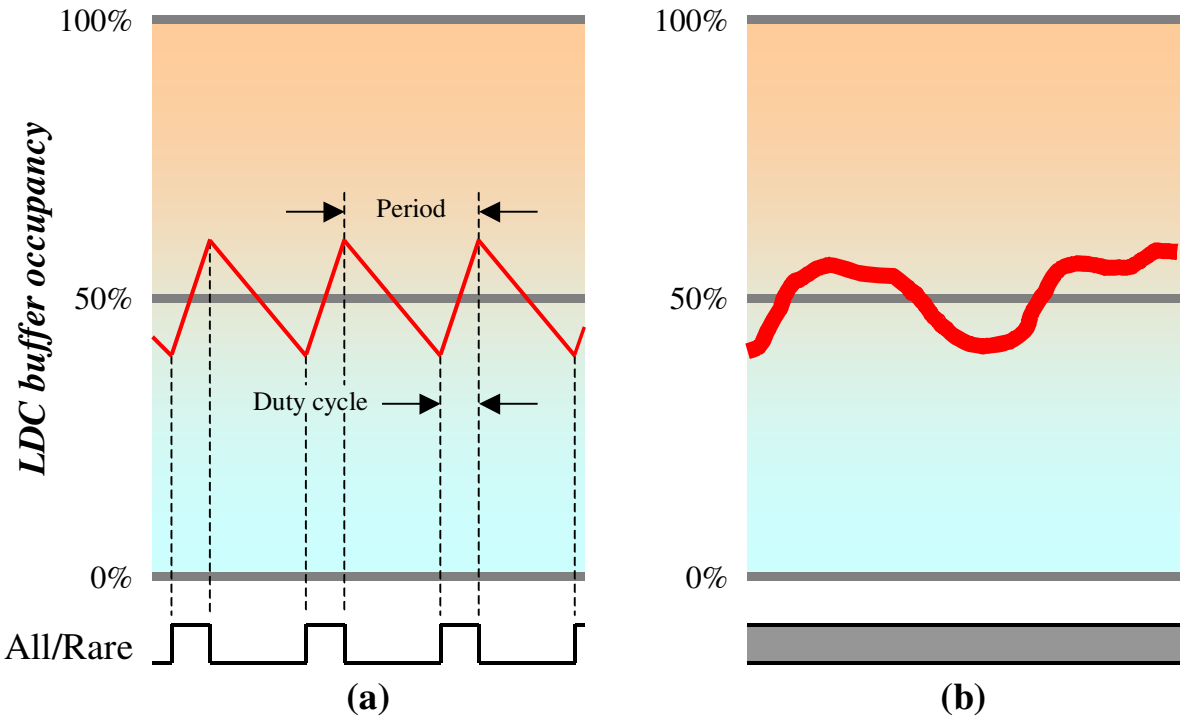

Figure 3 The All/Rare signal generated as a periodical function

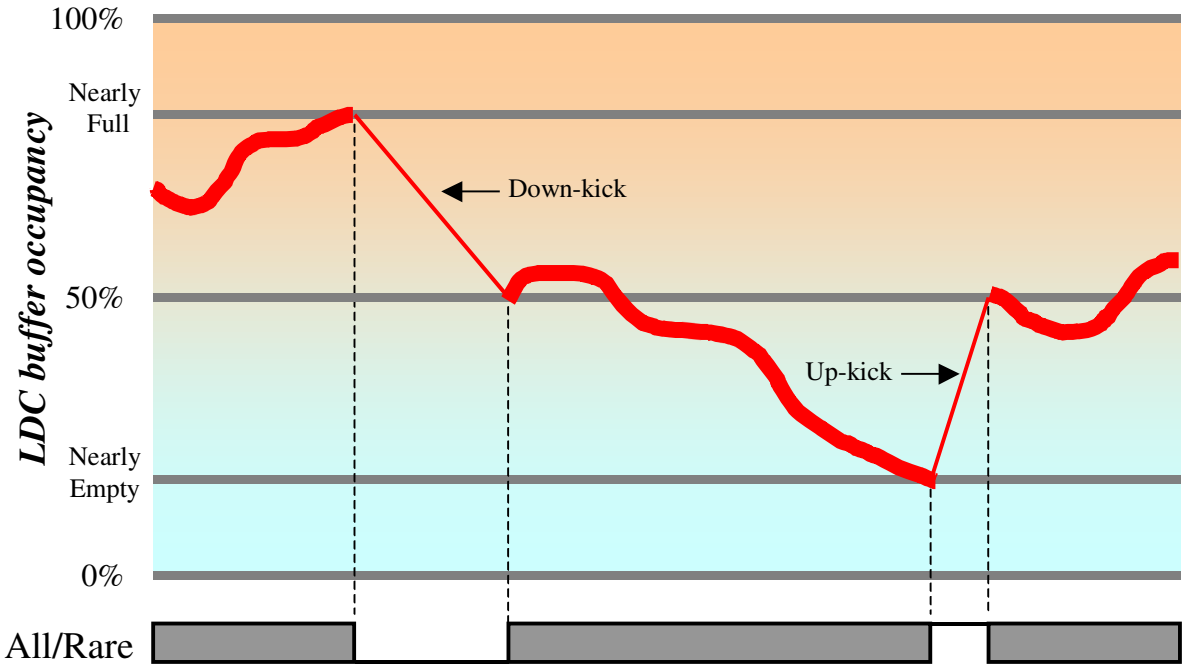

Figure 4 Servo monitors the *Nearly Full/Nearly Empty* limits

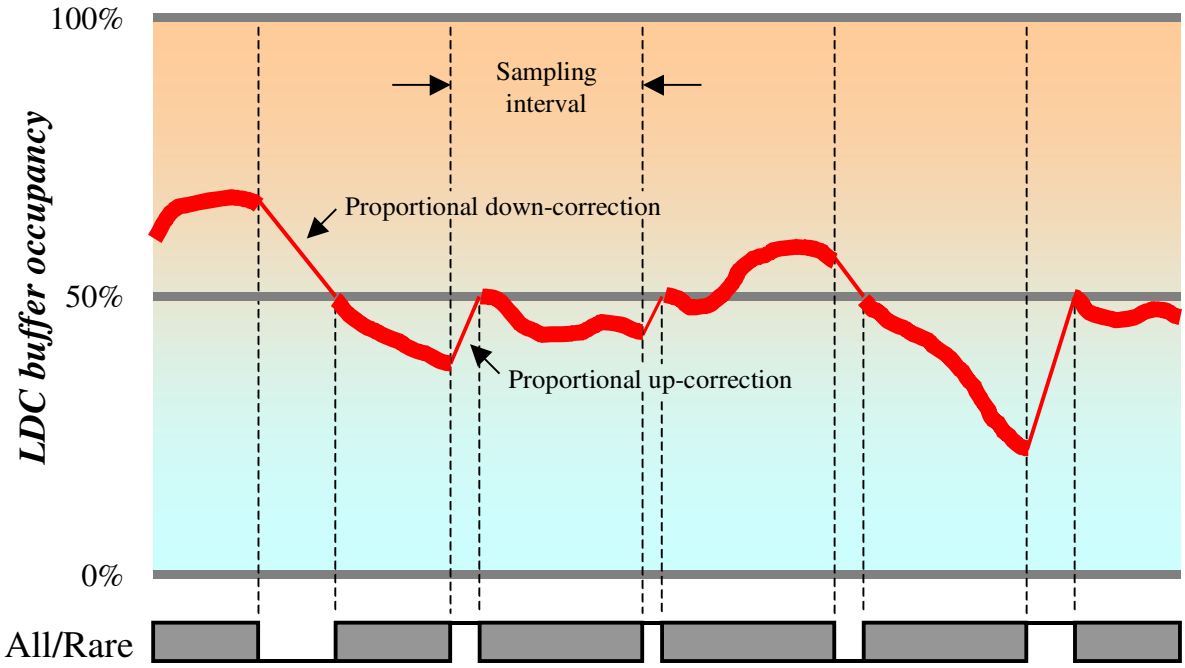

Figure 5 Servo based on periodical sampling

I'll repeat here the statement from the "proposal" that, *in all the cases*, "temporary dwelling in either status (buffer *full* of buffer *empty*) only makes the system operation less than optimal (*full* - looses the precious rare event acquisition time; *empty* - contributes to the inefficiency); *neither leads to a fatal error or a major degradation*. This simplifies considerably the task of servo control.

Different schemes involve the DAQ/HLD system in different ways and it's up to the DAQ/HLT group(s) to adopt the most appropriate one. *The decision is required relatively soon* since it affects the hardware implementation of the CTP.

2.1.3. Discard data in the LDC's

It would be possible to discard the data in the LDC's just after the transfer over the DDL's. However, this seems a waste of lifetime and bandwidth. It would also not be a big simplification over the previous scheme as it would require a synchronisation between all the LDC's to decide which events to discard.

3. Next Steps

The existing simulation program will be modified to include the proposed implementation. Different scenarios will be simulated to verify the behaviour.

I would like to add a couple of action items:

- Re-check the *cross section calculation* with the rare event scheme implemented. Is any additional hardware required (scalers, timers, *etc.*)? My "worry" has already been mentioned in the "proposal" (section ***Action***).
- Check the *effect of the past-future protection* (interaction rate) on the acquisition of rare events. In the current design, the protection is associated with clusters; if the protection requirements for rare events need to be relaxed, more clusters might be required, which is a very critical issue and might require (again) a complete re-think of the system.

Please comment!