Testplan for the calibration of the DFM component

Authors : S. Wheeler
Institute : University of Alberta, Edmonton, Alberta, Canada

Keywords : Model, Ptolemy, DFM

Abstract

Draft testplan for calibration of DFM application with regard to the Ptolemy model.

NoteNumber :
Version : Draft for discussion
Date : 22 April 2002
Reference : http://www.hep.ucl.ac.uk/atlas/simulation/

1 Introduction

The current Ptolemy model of the Data Flow Manager (DFM) is based on the description of the design of the actual application described in [1]. The different messages a DFM node can send and receive are summarised in Figure 1. The names used for the messages are the same as those used in the Message format note [2]. [1] The rationale behind the Ptolemy model is to treat the DFM as a "black box", rather than a detailed model of the application with detailed knowledge of the internal timings of the DFM. By defining a series of tests to calibrate the DFM, rather than relying on results of internal measurements we can decouple ourselves from changes in the internal structure of the application which would imply the updating of internal measurement hooks. Provided the same tests can be run, the model can be easily re-calibrated.

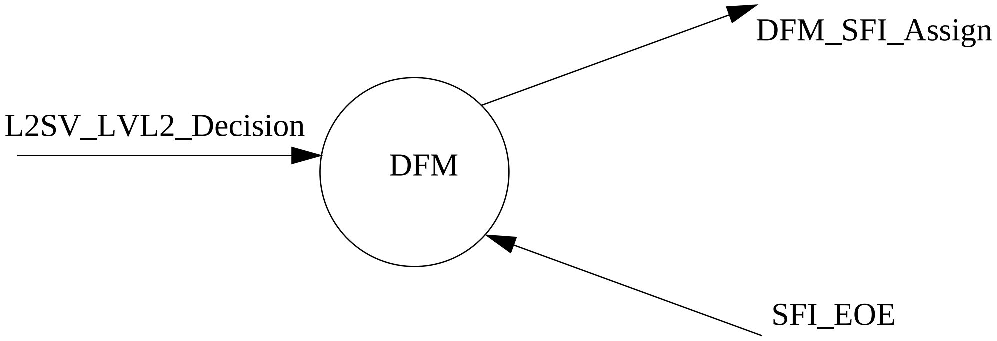

Figure 1: Input and Output Messages for the DFM

Although based on the design described in [1], the model of the DFM has been simplified by making the assumption that network I/O times will be much greater than any internal operations within the application. We also assume that only "normal" steady-state running conditions are to be modelled, therefore timeouts and the reception and handling of SFI_Busy, SFI_Non_Busy messages can be disregarded. With these assumptions in mind we propose that the DFM can be successfully characterised using the following 3 timing constants:

- time_to_receive_LVL2
- time_to_send
- time_to_receive_EoE

where time_to_receive_LVL2 is the total time spent (both in hardware and software) to collect from the network and analyse the L2SV_LVL2_Decision message received from a LVL2 Supervisor, time_to_send is the total time required to create and broadcast the DFM_SFI_Assign message to the ROS(es) and time_to_receive_EoE is the total time required to collect from the network and analyse the SFI_EOE message sent from an SFI.

[1] Question: For consistency should we consider changing the names of the messages used in the Ptolemy model to be the same as those defined in [2]?

Figure 4 in [1] shows the sequence diagram for receiving and processing a L2SV_LVL2_Decision message and sending the resulting DFM_SFI_Assign message. In the Ptolemy model the sequence diagram is considerably simplified and is shown in figure 2.

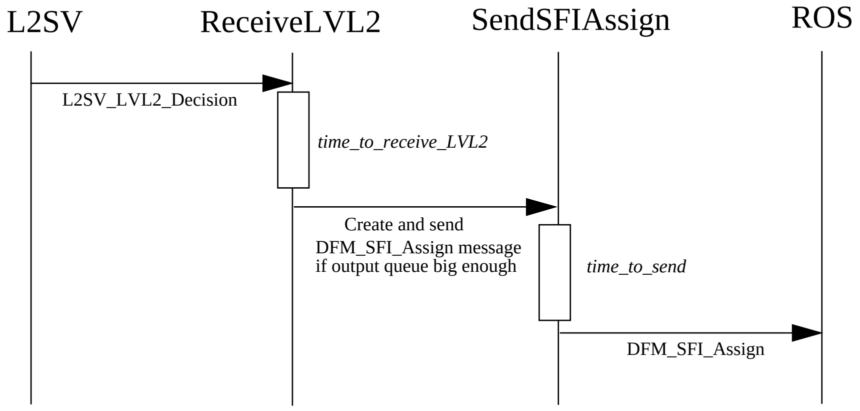

Figure 2: Simplified sequence diagram depicting the handling of a L2SV_LVL2_Decision message

Referring to figure 4 in [1], time_to_receive_LVL2 will include the time taken to receive the L2SV_LVL2_descision from the network and the time spent in the DFM classes, MessageDispatcher and L2DecisionGroupHandler. time_to_send will include the time spent in the classes LoadBalancer and DFMOutputHandler and the time to send the DFM_SFI_Assign message out onto the network.

Figure 5 in [1] shows the sequence diagram for processing an SFI_EOE message. The simplified diagram used by the Ptolemy model is shown in figure 3.

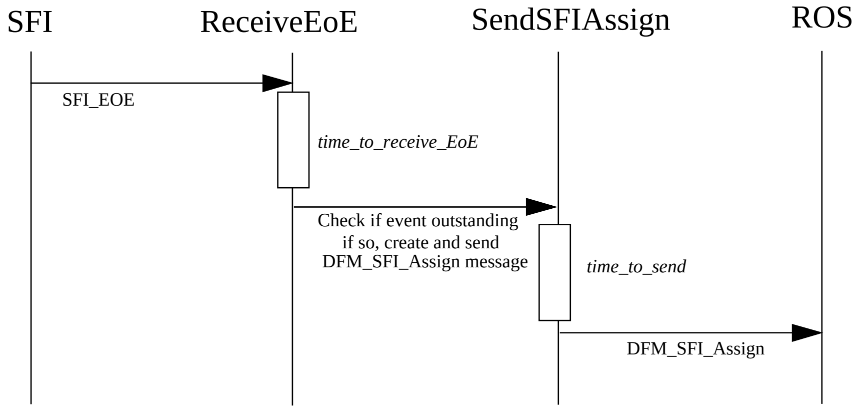

Figure 3: Simplified sequence diagram depicting the handling of an SFI_EOE message

time_to_receive_EoE will include the time taken to receive the SFI_EOE message from the network and the time spent in the DFM classes, MessageDispatcher and SFIControlHandler. time_to_send will include the time spent in the classes LoadBalancer and DFMOutputHandler and the time to send the DFM_SFI_Assign message out onto the network. A DFM_SFI_Assign message is only sent at this point if a LVL2 accepted event is outstanding i.e. all SFIs had reached their event limit when a previous attempt was made to assign the event to an SFI. We are assuming that during normal running the event buffers of the SFIs will be sufficiently large that this case will not normally occur (according to discussions with HPB). However, this feature is exploited in some of the calibration tests described later.

The purpose of this note is to describe the testbed measurements which need to be made in order to calibrate the Ptolemy model of the DFM, i.e. find values for the above times for a given (or possibly a number of) network protocol(s).

We are assuming that a scheme similar to that used to calibrate the model of the LVL2 Supervisor application on the Ethernet Testbed, see [3] can be used here. In addition to measurements made for calibration purposes, a number of other measurements must be made so that comparisons can be made between these measurements and the values predicted by the model once calibrated. For the time being, measurements and calibration will only be done for the DFM running the "classic" event building scenario, see [4].

All the calibration measurements should be made on a minimal Event Building system comprising, 1 DFM, 1 ROS and 1 SFI with one or many L2SVs (LVL2 Supervisors) or equivalent to inject L2SV_LVL2_Decision messages into the system, see figure 4. The basic aim of the tests is to saturate the component of the system under study. The time spent in the component under study can then be easily derived from the overall event rate and used in subsequent calculations.

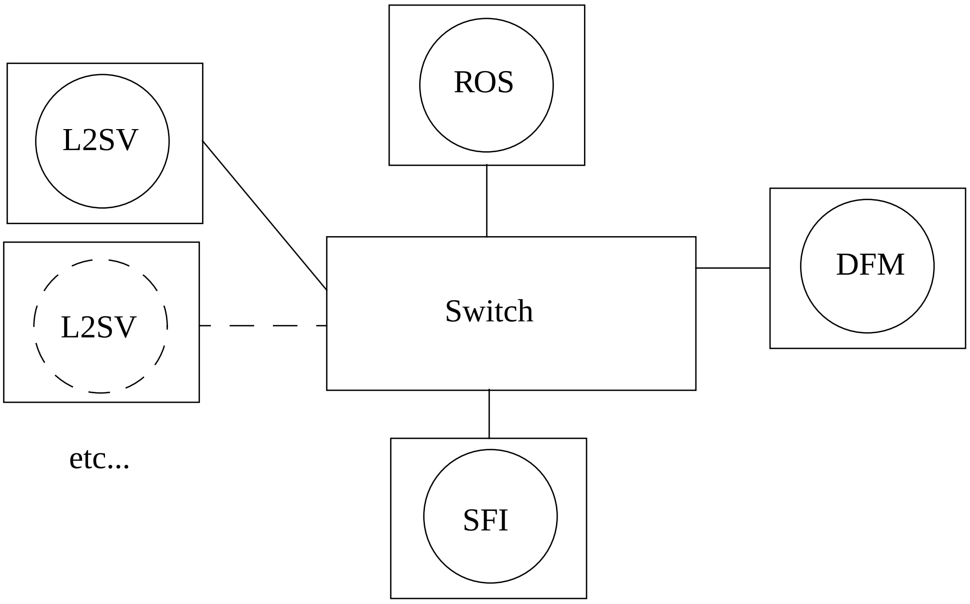

Figure 4: Basic configuration for DFM calibration measurements. Circles represent processes, squares, the hosts on which they run

Implications for the DataCollection Software:

- **SFI:** should be configured to expect only 1 event fragment of constant size per event
- **DFM:** should be able to measure global event rate, i.e. total number of SFI_EOE messages received divided by total running time

Although in the “classic” scenario the DFM broadcasts the DFM_SFI_Assign message to all the ROSes, in the simple tests described below, only a single ROS is used so at the moment this is not an issue.

Finally, the scheme proposed in this document for determining the various timing parameters is based entirely on saturation measurements of the various components in “realistic” setups, i.e. each component under test communicates with real applications rather than specially written test applications. Complementary tests could include the use of special tester applications and the use of internal timing measurements [5] to determine the timing parameters. Cross-checks between the different methods should provide confidence in the values of the parameters obtained. However, a final verification must involve running a number of larger setups, measuring event rate and latency and then comparing these values with those obtained running the Ptolemy model using the parameters derived from the calibration measurements, see section 9 (still to be written!).

2 Note on Packing Factors

It is assumed that the rejection rate of LVL2 will be 99% and that a packing rate of the order of 100 will be used in the L2SV_LVL2_Decision messages in the final system. Each message will therefore contain < ~1 events accepted by LVL2, for which event building has to take place. However, it should be noted that if the L2SV_LVL2_Decision message is to be kept within one Ethernet frame, i.e. 1500 bytes, no more than ~180 LVL2 decisions can be packed into a single message (see [2] for message formats and lengths). For the current set of measurements this limit will be respected, otherwise when using raw Ethernet or UDP, the application would have to spend time assembling complete messages from multiple frames. If TCP is used the implication would be greater times spent in the TCP interface.

In order to work with the simplest system possible for calibration purposes, tests 1-3 (see below) we should use L2SV_LVL2_Decision messages with a packing rate of 1 (i.e. contain one event only) and this should always be an event accepted by LVL2. The packing factor used for the DFM_SFI_Assign messages will be set equal to that used for the L2SV_LVL2_Decision message so that only 1 DFM_SFI_Assign message is sent for each L2SV_Decision message. This is important for understanding the system with only 1 accepted event in the system. Referring to the sequence diagrams in figures 2 and 3 and the corresponding figures in [1] the parameter time_to_send should be almost exactly the same for both scenarios i.e. there is no need to copy a vector of LVL2 rejects in the receive LS2V_LVL2_Decision scenario (although this time is probably not significant compared to the time required to send the message over the network).

The effect of packing rate (up to a maximum of ~180 LVL2 decisions per L2SV_LVL2_Decision message), keeping the packing rate equal for L2SV_LVL2_Decision messages and DFM_SFI_Assign messages is investigated in test 4. Changes in the saturation rate of the DFM would be due to a combination of the increased overhead receiving the message from the network and possibly the copying of vectors of events within the DFM application itself. The measurements should show how time_to_receive_LVL2 changes as a function of packing factor. The effect of increasing the packing factor of the DFM_SFI_Assign in terms of multiples of the packing factor of the L2SV_LVL2_Decision message are investigated in test 5 and should show how the parameter time_to_send changes as a function of packing factor.

Implications for the DataCollection Software:

- **L2SV:** must be possible to change the number of LVL2 accepts per L2SV_LVL2_Decision and the total number of LVL2 decisions per L2SV_LVL2_Decision.
- **DFM:** must be possible to change the total number of events per DFM_SFI_Assign to be changed

3 Test 1: DFM Saturation

The aggregate time spent in the DFM (aggregate_DFM_time) is given by:

```
time_to_receive_LVL2+time_to_send+time_to_receive_EoE
```

and can be calculated as the reciprocal of the event rate measured by the DFM (i.e. total number of SFI_EOE messages received divided by total running time) if it never becomes idle.

This may be achieved by:

- sending L2SV_LVL2_Decision messages at a rate no less than 1/(time_to_receive_LVL2 + time_to_send). This means that once the DFM has sent a DFM_SFI_Assign message there should always be another L2SV_LVL2_Decision message waiting to be handled, if an SFI_EOE message has not arrived in the meantime. To achieve this a sufficiently large number of LVL2 Supervisors (or some other message generator which can send L2SV_LVL2_Decision messages at a configurable rate, if the event builder tests are to be run in isolation) should be used to feed L2SV_LVL2_Decision messages to the DFM at the required rate.
- the value used in the load balancing algorithm of the DFM for the maximum number of events that may be buffered by the SFI should be large so that the DFM never has to wait for an SFI_EOE message before assigning a new event to the SFI.
- the total time spent in the ROS and SFI to build the event (i.e. between the ROS receiving the DFM_SFI_Assign message and the SFI sending the SFI_EOE message) should be minimised compared to the time: time_to_receive_LVL2 + time_to_send in the DFM. Therefore, the ROS should send a 0 length ROS_Event_Fragment message to the SFI. The ROS and SFI should be run on powerful processors, compared to the DFM processor to minimise response time. It is assumed that response time is a linear function of CPU power.

A series of measurements should be made using the testbed configuration shown in figure 4 varying the quantities described above until saturation of the event rate is observed.

Suggested CPU powers (indicative only) for the different applications would be:

- L2SV, ROS, SFI: 800 MHz
- DFM: 400 MHz

Number of outstanding events per SFI: many

Number of LVL2accepts | LVL2rejects per L2SV_LVL2_Descision: 1 | 0

4 Test 2: ROS saturation

Note: The aim of the next 2 tests (numbers 2 and 3) is simply to find out the amount of time spent in the ROS and SFI as input into the calculation described in section 6. They are not meant to be used to calibrate these nodes. Another method for finding these times is to instrument the ROS and SFI to measure the times directly.

The aim of the next set of measurements is to run the system where the ROS is the system bottleneck, always busy processing events. The maximum event rate achievable is limited by the time it takes to process an event in the ROS (time_in_ROS).

In order to saturate the ROS:

- the input of DFM_SFI_Assign messages to the ROS should be sufficiently high that the ROS is always kept busy. The rate at which LS2V_LVL2_Decision messages are sent to the DFM should be no lower than 1/time_in_ROS and the maximum number of events outstanding allowed for the SFI should be increased from 1 until saturation of the event rate is observed.
- the time spent in the ROS to process an event should be much greater than that spent in either the DFM or SFI, so that the event rate is dominated by the time taken in the ROS. The DFM and SFI should be run on high powered processors and the ROS on a low powered CPU.

Suggested CPU powers (indicative only) for the different applications would be:

- L2SV, DFM, SFI: 800 MHz
- ROS: 400 MHz

Number of outstanding events per SFI: 1-many

Number of LVL2accepts | LVL2rejects per L2SV_LVL2_Descision: 1 | 0

5 Test 3: SFI saturation

The aim of the next set of measurements is to run the system where the SFI is the system bottleneck, always busy processing events. The maximum event rate achievable is limited by the time it takes to process an event in the SFI (time_in_SFI).

In order to saturate the SFI:

- the input of ROS_Event_Fragment messages to the SFI should be sufficiently high that the SFI is always kept busy. The rate at which LS2V_LVL2_Decision messages are sent to the DFM should be no lower than 1/time_in_SFI and the maximum number of events outstanding allowed for the SFI should be increased from 1 until saturation of the event rate is observed.
- the time spent in the SFI to process an event should be much greater than that spent in either the DFM or ROS, so that the event rate is dominated by the time taken in the SFI. The DFM and ROS should be run on high powered processors and the SFI on a low powered CPU. It is assumed that response time is a linear function of CPU power.

Suggested CPU powers (indicative only) for the different applications would be:

- L2SV, DFM, ROS: 800 MHz
- SFI: 400 MHz

Number of outstanding events per SFI: 1-many

Number of LVL2accepts | LVL2rejects per L2SV_LVL2_Descision: 1 | 0

6 Calculation of times in DFM

Assuming a configuration can be found, with only one event in the system, where none of the components are saturated and a LVL2 accept is always available the following calculations can be made. The sequence of operations within the DFM is the following (cf. section 10.1 of [3]):

- DFM receives SFI_EOE message from the SFI
- DFM spends time_to_receive_EoE to collect/analyse the message
- an L2SV_LVL2_Decision message has already been received but could not be sent because the number of outstanding events for the SFI was exceeded. The rate at which L2SV_LVL2_Decision messages are sent to the DFM must be sufficiently high that there is always a L2SV_LVL2_Decision message waiting to be handled.
- it now can be sent and the DFM takes time time_to_send to send it
- message traverses network and switch to ROS, latencies on FastEthernet and inside the switch are fixed and known for a given length of message. This can be called net_time1.
- ROS spends time time_in_ROS (Test 2) before sending the ROS_event_fragment message to the SFI
- ROS_event_fragment message takes net_time2 to traverse network to SFI
- SFI spends time time_in_SFI (Test 3) before sending the SFI_EOE message back to the DFM
- SFI_EOE message takes net_time3 to traverse network to DFM
- total roundtrip time (total_time) measured as 1/event rate

So total time event takes outside DFM (in the network, ROS and SFI) is:

```
time_out_of_DFM = net_time1+net_time2+net_time3+time_in_ROS+time_in_SFI
```

then:

```
time_to_receive_EoE+time_to_send = total_time-time_out_of_DFM
```

and:

```
time_to_receive_LVL2=aggregate_DFM_time-(time_to_receive_EoE+time_to_send)
```

Need some way to split time in a sensible way between time_to_receive_EoE and time_to_send. It is hoped that measurements to determine effect of message length suggested in Tests 4 and 5 will help.

7 Test 4: Investigation of message length on time to receive and send parameters

Following discussions with HPB the optimum packing factor(s) for the L2SV_LVL2_Decision and DFM_SFI_Assign messages have yet to be determined but will probably be of the order of hundreds of events, where most are rejected events (it is a parameter for which modelling should be able to give a value). The packing factor for the DFM_SFI_Assign message is likely to be equal to or greater than the packing factor used in the L2SV_LVL2_Decision message. In principle both the time_to_receive_LVL2 and time_to_send parameters will vary as a function of packing.

Tests designed to saturate the DFM (using similar configuration to Test 1) should be run to see how the aggregate_DFM_time varies for different packing factors (for simplicity each L2SV_LVL2_Decision whatever its length should only contain 1 LVL2 accept so that the number of events in the system is the same as before). Suggest we look at packing factors of 1, 100 and 184 (to keep within 1 Ethernet frame) and repeat with the DFM running on different powered CPUs (provided it stays saturated). The packing factor for DFM_SFI_Assign should be the same as for L2SV_LVL2_Decision. So that a DFM_SFI_Assign message is sent for each L2SV_LVL2_Decsision message received.

According to [2] the L2SV_LVL2_Decision consists of a generic header consisting of 6 words and then a list of LVL2 decisions, with 2 words per decision. We are assuming 4 bytes per word. There is a small (constant) amount of additional header information from Ethernet itself which is not included below because I do not know how much it is!

Packing factor of 1 gives message length 6+(1*2) = 8words/32 bytes

Packing factor of 100 gives message length 6+(100*2) = 206 words/824 bytes

Packing factor of 184 gives message length 6+(184*2) = 374 words/1496 bytes

Suggested CPU powers (indicative only) for the different applications would be:

- L2SV, SFI, ROS: 800 MHz
- DFM: 400 - 800 MHz

Number of outstanding events per SFI: many

# LVL2accepts | LVL2rejects per L2SV_LVL2_Descision (packing factor 1): 1 | 0

# LVL2accepts | LVL2rejects per L2SV_LVL2_Descision (packing factor 100): 1 | 99

# LVL2accepts | LVL2rejects per L2SV_LVL2_Descision (packing factor 184): 1 | 183

8 Test 4: Investigation of message length on time to send parameter

Tests designed to saturate the DFM and determine the total event rate should be run to see how the aggregate_DFM_time varies for different packing factors of the DFM_SFI_Assign message. The packing factor for DFM_SFI_Assign message should be an integer, N, multiple of the packing factor for the L2SV_LVL2_Decision message (each L2SV_LVL2_Decision message contains only 1 LVL2 accept). For each DFM_SFI_Assign message sent, N L2SV_LVL2_Decision messages and N SFI_EOE messages will be received. Therefore, the aggregate time spent in the DFM will now be:

```
(time_to_receive_LVL2+time_to_receive_EoE)*N+time_to_send
```

Suggest we look at packing factors of 200 (N=2) and 300 (N=3) for the DFM_SFI_Assign message and fixed packing factor of 100 for the L2SV_LVL2_Decision message and repeat with the DFM running on different powered CPUs (provided it stays saturated).

The DFM_SFI_Assign consists of the generic header consisting of 6 words, a header for the LVL2 accepts consisting of 2 words, then 3 words per LVL2 accept, a padding word if required, then a header for the LVL2 rejects consisting of 2 words, then 1 word per LVL2 reject and a padding word if necessary. Again overhead from Ethernet is excluded.

Packing factor 200: 6+2+(3*2)+2+198 = 214 words/856 bytes

Packing factor 300: 6+2+(3*3)+1+2+297+1 = 318 words/1272 bytes

Suggested CPU powers (indicative only) for the different applications would be:

- L2SV, SFI, ROS: 800 MHz
- DFM: 400 - 800 MHz

Number of outstanding events per SFI: many

# LVL2accepts | LVL2rejects per L2SV_LVL2_Descision (packing factor 100): 1 | 99

9 Other Measurements to compare with model once calibrated

1 DFM, 1 ROS, 1-several SFIs, 1-several LVL2 accepts/DFM_SFI_Assign? more needs to be written here...sorry

10References

1. The DFM Design, C.Häberli, DataCollection Note 23, Version 0.3, 22 Jan 2002
2. The Message Format used by DataCollection on the ATLAS TDAQ Integrated Prototype, HP.Beck, F.Wickens, DataCollection Note 22, Version 0.8, 25 Jan 2002
3. Ptolemy simulation of the ATLAS level-2 trigger, P.Clarke et al., ATL-DAQ-2000-039
4. High-Level Description of the Flow of Control and Data Messages for the ATLAS TDAQ Integrated Prototype, HP.Beck, C.Häberli, DataCollection Note 12, Version 0.2, 25 Oct 2001

5. Calibration Results for the DFM. S.Gadomski, in preparation