

# Investigation of the Networking Performance of Remote Real-time Computing Farms for ATLAS Trigger DAQ

Bryan Caron, Richard Hughes-Jones, Krzysztof Korcyl, Catalin Meirosu, Jakob Langgard Nielsen

**Abstract**— To test the feasibility of using remote farms to perform real-time event selection in the Trigger/Data Acquisition System for the ATLAS experiment at CERN, a Proof of Concept was set up during 2004. The behaviour of the request-response protocol to move application data has been measured for remote farms connected with different Wide Area Networks including a dedicated lightpath, a Virtual Private Network, and the standard production network. The dynamics and effect of using TCP/IP as the transport protocol has also been investigated for this real-time application. These data are compared with conventional bulk data transfers and used to validate the observed performance of the online farms and estimate the effect of this traffic pattern on the Wide Area Network.

**Index Terms**— TCP, Protocols, real time systems, wide area networks

## I. INTRODUCTION

Several experiments, including ATLAS at the Large Hadron Collider (LHC) and D0 at Fermi Lab, have expressed interest in using remote computing farms for processing and analysing the information from particle collision events. Different architectures have been suggested, ranging from pseudo-real-time file transfer and subsequent remote processing, to the real-time requesting of individual events described here.

To test the possibility of using remote computing farms for real-time processing within the ATLAS experiment, a collaboration [1] was set up between members of ATLAS Trigger/DAQ, Canarie, DARENET, Netera, PSNC, UKERNA and Dante to demonstrate a Proof of Concept and measure end-to-end network performance. The testbed shown in Fig. 1

Manuscript received June 13, 2005. This work was supported in part by the Particle Physics and Astronomy Research Council in the UK, by the European Union under the IST-2001-33185 grant and also by the Polish Funding Agency under the grants 620/E-77/SPUB-M/CERN/P-03/DZ 110/2003-2005 and 620/E-77/SPB/5\_PR UE/DZ 465/2002-2004.

B. Caron is with the The University of Alberta, Edmonton, Canada

R. Hughes-Jones is with The School of Physics and Astronomy, The University of Manchester, Oxford Rd., Manchester, M13 9PL, U.K. (Corresponding Author: R.Hughes-Jones, phone: +44 161 275 4117; fax: +44 161 273 5867; e-mail: [R.Hughes-Jones@manchester.ac.uk](mailto:R.Hughes-Jones@manchester.ac.uk))

K. Korcyl is with IFJ PAN Krakow, Poland.

C. Meirosu is with CERN, Geneva and with the “Politehnica” University of Bucuresti, Romania.

J.L. Nielsen is with Niels Bohr Institute, Copenhagen, Denmark

was centred at CERN and used three different types of wide area high-speed network infrastructures to link the remote sites:

- an end-to-end Ethernet over SDH lightpath to the University of Alberta in Canada
- standard end-to-end IP connectivity over the academic production network to the University of Manchester in the UK and the Niels Bohr Institute in Denmark
- a Virtual Private Network (VPN) composed of an MPLS tunnel over the GEANT network and an Ethernet VLAN over the Polish PIONIER network to IFJ PAN Krakow.

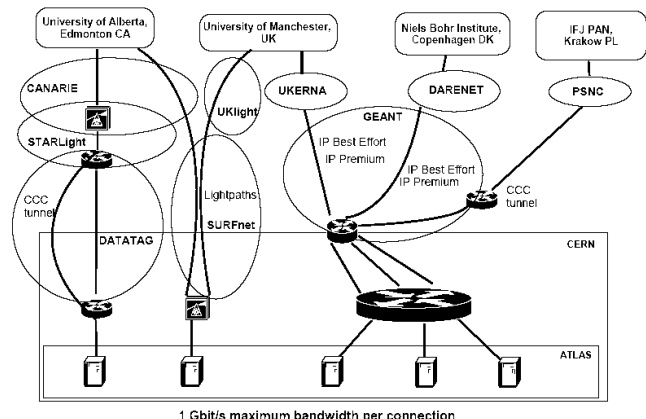


Fig. 1. The network configuration of the testbed

For potentially interesting particle interactions in ATLAS, the standard data acquisition system [2] records all the data fragments from the sub-detectors and builds them into full blocks of data (called events in the particle physics world) using code called the Sub-Farm Interface (SFI). These events are then fed to processors that form the Event Filter (EF) that analyse the detailed physics content of the event. Selected events are then recorded. As in the standard data acquisition system, the Proof of Concept used TCP/IP streams to move events from an SFI located at CERN to remote sites for Event Filter computation and then return the results. The remote EF processors used a request-response protocol to obtain the events from the SFI and return the result to the Sub-Farm Output (SFO). In the final experiment, a typical event will be ~1.5 MBytes and EF computation times of the order of 1-2 s.

To meet security requirements [3], the testbed at CERN used a private network. This required the use of Network Address Translation (NAT) systems to allow connections to be made to the remote farms over the production network. The remote farms connected using a lightpath or a VPN appeared as an extended LAN and used the same IP network addresses as the systems at CERN.

A comprehensive discussion on potential scenarios using remote computing farms with the current ATLAS Trigger/DAQ software is reported elsewhere [4].

## II. METHODOLOGY

### A. End Hosts, NAT systems, NICs and Networks

Even though server-quality PCs (the PCs used SuperMicro P4DP8 motherboards) were used for these tests, it was important to characterise the performance of the end systems, the NAT boxes, as well as the networks involved. The end hosts and NAT boxes should have sufficient CPU power, memory bus bandwidth and Input/Output (I/O) capability to cope with the expected network traffic, i.e. packets should not be dropped in the end host itself. It was also important to establish the throughput and packet loss of the end-to-end network infrastructures used.

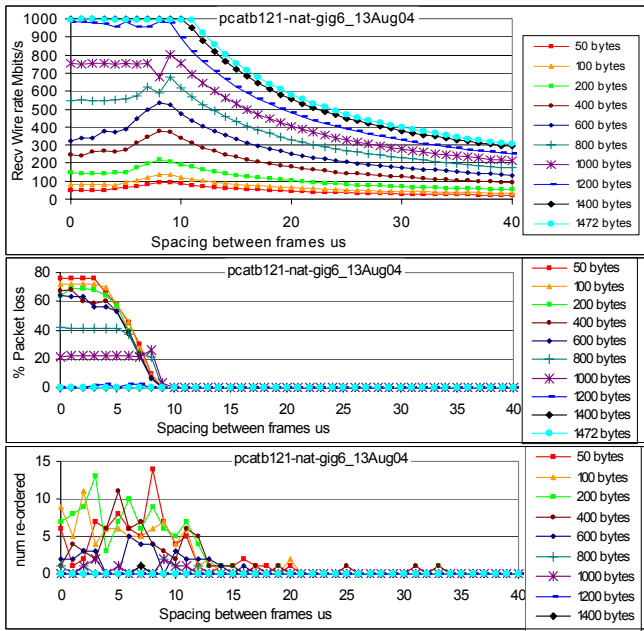


Fig. 2. Results of UDP traffic streaming through the NAT box

A methodology [5] for evaluating the end host network performance by using UDP packets to measure the latency, throughput, jitter, packet loss and the activity on the PCI/PCI\_X buses was used to evaluate these PC systems. UDPmon [6] was used to send streams of UDP packets spaced at regular, carefully controlled intervals between the server systems connected back to back. The methodology was also applied to each of the networks tested to measure end-to-end performance and characterise packet loss.

Fig. 2 shows the UDP memory-to-memory tests between PCs at CERN and Manchester through the NAT box. For packets greater than 1200 bytes the systems were capable of operating at line speed with no packet loss, but small packets suffered considerable losses. However, this did not affect the TCP flows as the small packets used by TCP to acknowledge the data were only sent every other received data packet ( $\sim 24 \mu\text{s}$ ) where there is no observed loss. The packet re-ordering shown in the bottom graph in Fig. 2 was due to the multiple forwarding engines in the core routers of the production network.

### B. The Request-Response Application Protocol

The data transfer protocol used in the TDAQ DataFlow application is presented in Fig. 3. The EF sends a small request to the SFI, which immediately responds by transferring an entire event. The EF processes the event data and, if the event is considered interesting from the physics point of view, requests a buffer for temporary storage at the SFO. The SFO grants the request (immediately, if enough space is available) and as soon as the EF receive this grant response, it starts transferring the event to the SFO. The work presented in this paper focused on determining the performance of the transfers over a long-distance network, so all the data was dummy and no physics analysis was performed.

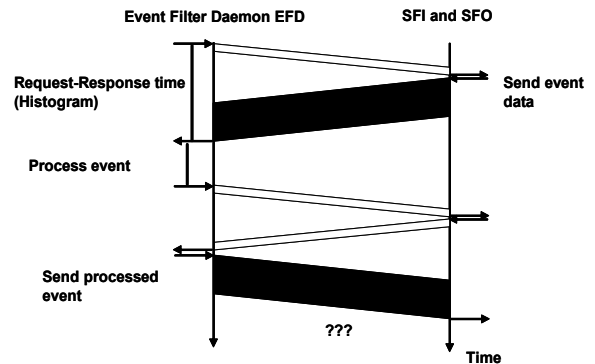


Fig. 3. The request-response protocol of the ALTAS Event Filter

The operation of the event request-response application protocol used in Trigger/DAQ was investigated with the test program tcpmon [7]. Tcpmon implemented a request-response protocol over a TCP connection to transfer data as shown in the top portion of Fig. 3. This program was instrumented in a similar way to udpmon and provided histograms of the round trip request-response times as well as a time series of these latencies.

### C. TCP Stacks and Operation

Work on advanced network protocols implementing sender side modifications to TCP highly increased bandwidth utilisation in long delay high bandwidth and multi-user network environments [8]. This allows a single stream of a modified TCP stack to transmit at rates that would otherwise

require multiple streams of standard TCP. The measurements reported here were made on systems using RedHat Linux 9 with the 2.4.20 kernel, patched to allow choosing the active TCP stack algorithm[9] with no system reset required after changing the stack. The possible choices included standard TCP, High Speed TCP and Scalable TCP. The High Speed and Scalable stacks both make the reduction of the rate less severe when detecting packet loss, whilst increasing the transmission rate more aggressively than standard TCP during the recovery. Web100 [10] was used to instrument the TCP stacks.

### III. RESULTS AND DISCUSSION

#### A. The Request-Response Protocol over the WAN

After characterising the end to end links using UDPmon, tcpmon was used to investigate the performance of the request-response traffic. In these tests, the size of the TCP send and receive buffers were set to delay\*bandwidth product for each link used.

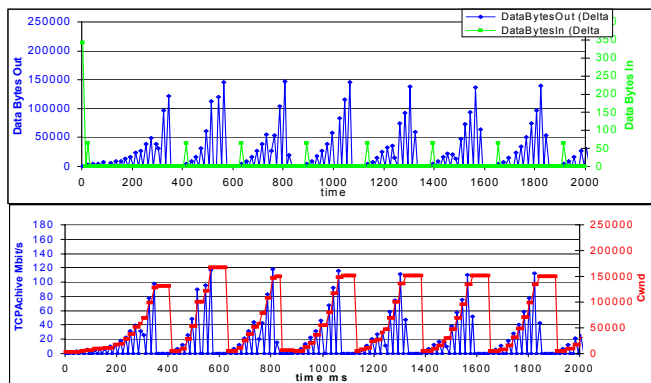


Fig. 4. Request-response traffic showing the effect of automatic cwnd reduction. Upper plot: the request is shown in green, response in blue Lower plot: The variation of cwnd in red and the TCP throughput in blue.

Fig. 4 shows the behaviour of the standard TCP stack for the first 2 seconds of the test when tcpmon was run between Manchester and CERN. A new request was sent 80ms after each response was received. The upper plot shows the 64 byte requests in green, followed by the 1 Mbyte response in blue. TCP is in its slow start phase where it gradually increases the amount of data sent, in this case it takes 8 round trip times (RTT) or ~160 ms to send the response. The blue points in the lower plot show the TCP throughput and the red line indicates the behaviour of the TCP Congestion Window (cwnd) as a function of time. After the pause in sending data (which would occur while the remote CPU was analysing the event), the TCP stack dramatically reduces the congestion window so TCP is always in the slow start phase limiting the request-response latency to 160ms and the maximum throughput to ~120 Mbit/s. Fig. 5 shows the case when the Linux TCP cwnd reduction implementation feature has been turned off. TCP slow start allows the congestion window to open, when it does, the response occurs in about 1-2 rtt giving a request-

response latency of ~40ms and a maximum throughput of ~800 Mbit/s.

Fig. 6 shows the results of the same test, performed using a tuned TCP stack, on the Geneva-Edmonton connection. Having an RTT of 150 ms, this was the longest connection of our testbed, giving a correspondingly longer TCP slow start period. The first 1 Mbyte event took about 1.8 seconds. Soon afterwards the TCP stack entered the congestion avoidance phase and the throughput grew slowly up to about 800 Mbit/s.

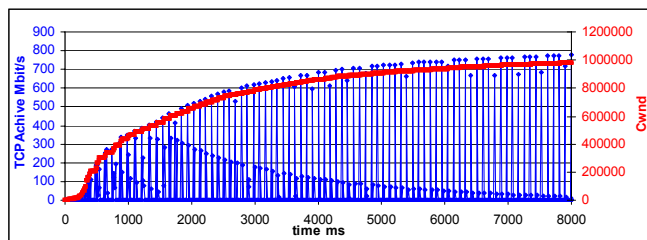


Fig. 5. The bandwidth of the request-response traffic with tuned TCP stack

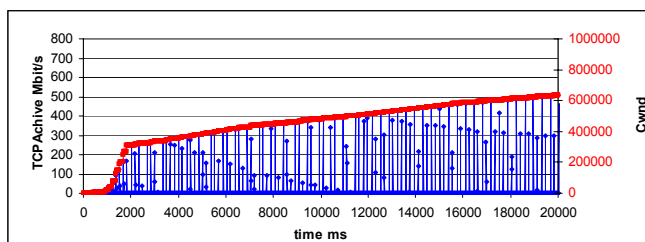


Fig. 6. Request-response transfers on the Geneva-Edmonton connection

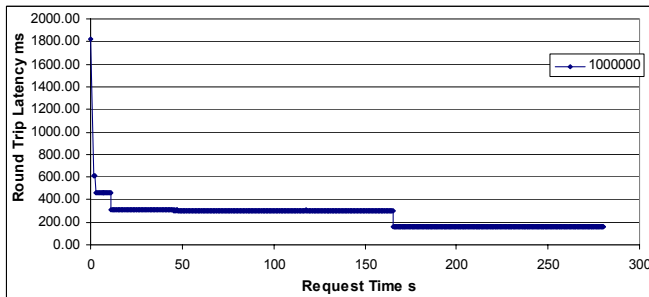
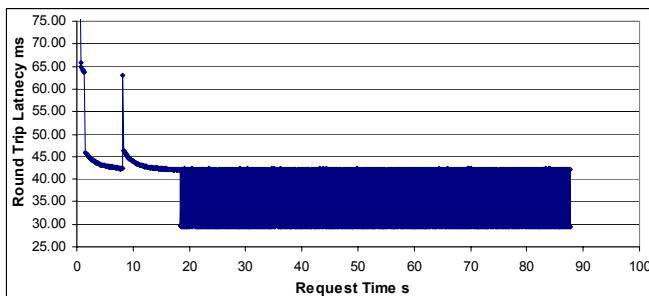


Fig. 7. The delay of the request-response transaction on the Geneva-Manchester (a) and Geneva-Edmonton (b) connections

Fig. 7 presents the time series plots of the individual request-response transaction latencies obtained on the Geneva-Manchester and Geneva-Edmonton connections, for a response of 1 MB in length. The delay is clearly dominated by the physical RTT of the respective connection, as demonstrated in the above discussion. The artifacts present in

the graphs, such as the alternate latencies of 29 ms and 42.5 ms on the Geneva-Manchester plot and the step changes in the latency observed on the Geneva-Edmonton circuit, still have to be understood.

Fig. 7 also illustrates the uncertainties faced when transmitting small amounts of data over long-distance connections, sharing part of or the entire end-to-end path with other users.

### B. Using the ATLAS DataFlow Application.

As one of the tests, the Atlas Online and DataFlow software[11] was configured at CERN to operate a three node EF farm in Manchester, while the SFI and SFO applications ran on the same node installed at CERN. Two of the EF nodes were connected through 100 Mbit/s Ethernet links, while the third node had a Gigabit Ethernet link to the same switch that was in turn connected to the wide area network. The ATLAS Online software allows for pre-defined nodes to be removed or added from/to the system configuration at runtime. The magenta line in Fig. 8 presents the number of active EFD nodes during a certain time interval and the solid points show the average total rate of received events, as reported by the SFO.

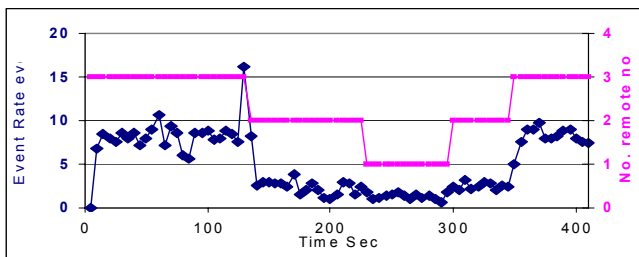


Fig. 8. Average event rate received by the SFO as a function of the time though the test.

At about 120s into the test, the node with the Gigabit Ethernet connection was removed and at about 220s another node was removed. At 300s and 350s the nodes were returned to the farm. The results show a rate of about 6 Hz, for the node connected through Gigabit Ethernet and 1-2 events/s for the 100 Mbit nodes. At this time the socket buffers of the application could not be set to the value of the delay\*bandwidth product as used in the other tests described above. The RTT was 20ms and for the transferred event size of 1 MB the transfer time determined by tcpmon was 42.5 ms. Taking into account the SFI-EF-SFO communication, the time for returning an event to the SFO would be around 95 ms. giving an expected event rate of 10.5 Hz.

Tcpdump, a software tool that allows the parameters of individual packets to be recorded, was used to investigate the behaviour of the application. Tcpdump was run on the SFI node at CERN. Fig. 9 and Fig. 10 show the data transfer from the SFO to the EF. Fig. 9 shows the classic TCP slow start period of the communication between the SFI and the EF, which took roughly 320 ms, and Fig 10 shows the detailed packet dynamics as a function of time for the transfer of a 1

Mbyte response, taken well after the slow start phase had ended. The almost vertical black lines are composed of small bars, each bar corresponding to a TCP packet of 1448 bytes. The green line represents the amount of transmitted data that has been acknowledged by the remote node. time line, with steps corresponding to the acknowledgements packets received by the TCP.

Fig. 10 shows that data are sent out in a small burst of packets sufficient to fill but not overflow the remote TCP buffer. As soon as the acknowledgements arrive, one RTT later, more data (represented by the black lines) are sent on the TCP connection. This plot demonstrates that the application could not set the buffer size of the receiving socket to the required value, thus TCP was unable to obtain the maximum transfer rate on the connection. It took about 115 ms to transmit one event, which meant a rate of ~4 events per second for the overall system, much lower than the expected 10 events per second and closer to the measured 6 events per second.

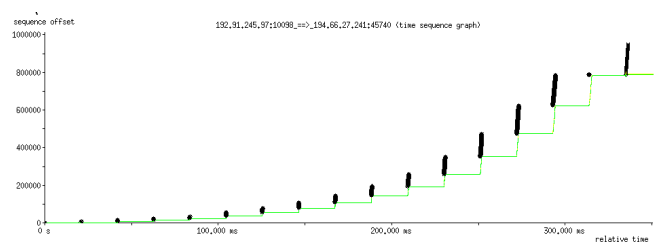


Fig. 9. Detailed view of the slow start phase of the SFI-EFD communication

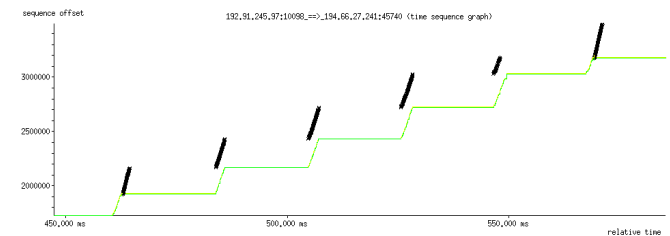


Fig. 10. Detailed view of the SFI-EF communication

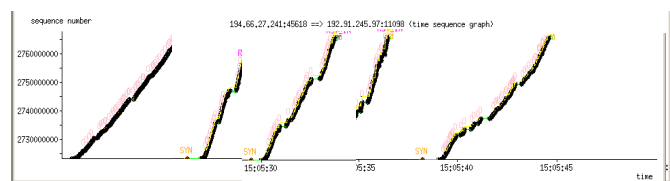


Fig. 11. tcpdump trace of the EFD-SFO connection

The SFO application establishes an application level timeout for the interval when a complete event should arrive. This was fixed at a value characteristic for a local computing cluster environment. The counting starts after sending a positive answer to the EF's request for temporary storage space. Due to the TCP buffer limitations discussed above, sometimes the event cannot arrive within the configured timeout value. The application considered this to be an error and dropped the connection voluntarily, only to try re-enabling it immediately. Fig. 11 shows this behaviour, when the dark lines terminate

near the top of the plot. Unfortunately, the newly established connection will have to pass through the TCP three way handshake and the slow start phases before being able to transmit data at the highest rate. As TCP has its own timeout mechanisms and can inform the application if a link is terminated, it is clear that the use of application timeouts and the response to error conditions must be approached with some care.

#### IV. CONCLUSION

We investigated the behaviour of an application designed for a computing cluster when used in a long-distance network scenario. The application consisted of a request-response protocol, running over a TCP/IP connection and designed for high-energy physics data analysis in the Trigger and Data Acquisition System of the future ATLAS experiment at CERN.

The dynamics of the TCP protocol strongly influence the performance achieved by the application. Due to the request-response nature of the protocol, it is not the TCP throughput but the round-trip time that determines the performance of the application. The transmission time of the first block of data sent after the opening of the connection is considerably increased by the TCP slow start. It is also essential to set the TCP buffer sizes to allow rapid transmission of the data. Linux TCP implementation-specific optimisations, like the automatic reduction of the current window size after a certain time of inactivity on the connection considerably reduced the performance of our application.

The request-response nature of the protocol under investigation makes this application different from standard high-energy physics applications deployed over long-distance networks, which usually involve the bulk transfer of large files. However, in addition to the high-energy physics field, our findings are relevant to applications that use iSCSI transfers over Internet remote database accesses or to interactive medical imaging applications that need to transfer large patient images in real time. These investigations are also relevant to simulations that use High Performance Computing facilities with a remote researcher requiring real-time visualisation to allow interactive computational steering using haptic input devices.

#### ACKNOWLEDGMENT

We would like to thank members of the ATLAS Online and DataFlow groups for their help in making this work possible. We would also like to thank Edoardo Martelli and Paolo Moroni from CERN as well as members of Canarie, DARENET, Netera, PSNC, UKERNA and Dante for their help in configuring the National Research Networks and GEANT.

#### REFERENCES

- [1] R. Hughes-Jones, B. Caron, K. Korcyl, C. Meirosu, A. Waananen "A Proof of Concept Demonstration of Remote Farms for Real-Time Processing for ATLAS Trigger/DAQ", ATLAS TDAQ, Jun 2004
- [2] ATLAS High-Level Trigger, Data Acquisition and Controls, Technical Design Report, CERN/LHCC/2003-022, July 2003.
- [3] Uwe Epting et al. "Computing and Network Infrastructure for Controls (CNIC) Policy", <http://wg-cnlic.web.cern.ch/wg-cnlic/>
- [4] C. Bee et al., "On the Potential Use of Remote Computing Farms in the ATLAS TDAQ System", 14<sup>th</sup> IEEE Real Time Conference, Stockholm, Sweden, June 2005
- [5] R. Hughes-Jones, P. Clarke, S. Dallison, "Performance of 1 and 10 Gigabit Ethernet Cards with Server Quality Motherboards," Future Generation Computer Systems Special issue, 2004
- [6] UDPmon: a Tool for Investigating Network Performance, <http://www.hep.man.ac.uk/~rich/net>
- [7] TCPmon: Test program Instrumenting the Request-Response Protocol. <http://www.hep.man.ac.uk/~rich/net/tcpmon>
- [8] H. Bullo, R. L. Cottrell, R. Hughes-Jones, "Evaluation of Advanced TCP Stacks on Fast Long-Distance Production Networks," Journal of Grid Computing, Volume 1, Issue 4, 2003, Pages 345 - 359
- [9] The 2.4.20 kernel was from <http://www.kernel.org/> patched with Yee Ting Li's patch 20040119\_linux-2.4.20-altAIMD-0.3-web100-2.3.3\_sacks.patch see [www.hep.man.ac.uk/~rich/SC2004/patches](http://www.hep.man.ac.uk/~rich/SC2004/patches)
- [10] Web100 Project home page <http://www.web100.org/>
- [11] A. dos Anjos (On behalf of the ATLAS High Level Trigger Group)"Deployment of the ATLAS High Level Trigger", proceedings 14<sup>th</sup> IEEE Real Time Conference Stockholm, Sweden , June 2005.



**Richard Hughes-Jones** leads the e-science and Trigger and Data Acquisition development in the Particle Physics group at Manchester University. He has a PhD in Particle Physics and has worked on Data Acquisition and Network projects. He is a member of the Trigger/DAQ group of the ATLAS experiment in the LHC programme, focusing on Gigabit Ethernet and protocol performance. He was responsible for the High performance, High Throughput network investigations in the European Union DataGrid and DataTAG projects, and the UK e-Science MB-NG project, he continues this role in the UK GridPP and ESLEA projects. He is secretary of the Particle Physics Network Coordinating Group which supports networking for UK PPARC researchers. He is a co-chair of the Network Measurements Working Group of the GGF, a co-chair of PFLDnet 2005, and 2006, and a member of the UKLight Technical Committee.



**Catalin Meirosu** obtained a M.Sc. in image processing and artificial intelligence from the "Politehnica" University of Bucuresti, Romania in 2000. He then became a research assistant with the Politehnica University, where he is now pursuing a Ph.D. Also in 2000, he started working at CERN for the trigger and data acquisition system of the ATLAS experiment. He was involved in two European R&D projects: SWIFT and ESTA. Catalin's research interests are in the area of high performance networking with applications to real time data acquisition systems.