# KEY COMPONENTS IN A NETWORKED DATA ACQUISITION SYSTEM

**Diarmuid Corry**
**ACRA CONTROL INC**

**ABSTRACT**

With the growing interest in networked data acquisition there has been a lot of focus on networked data acquisition systems. However, the requirements of a flight test instrumentation system go beyond networked DAU's. For example, a FTI network fabric has particular requirements for switches, time grandmasters, recorders, data servers and network terminals to the ground. This paper discusses these components and how they inter-operate in a single, fully networked system and discusses some FTI oriented requirements for same. Where relevant, we discuss the results of some experiments with network latencies; packet losses etc. and discuss some enhancements that can contribute to improved efficiency for flight test programs.

Keywords: Networked Data Acquisition, Data Recorder, Data Server, Grandmaster, FTI

## INTRODUCTION

The last four or five years have moved away from proprietary and closed systems for Flight Test Instrumentation (FTI) networks, towards more open and commercially based systems centering around Ethernet. This move affects the way flight test data acquisition is performed on every level – from the physical interfaces, through the configuration and management and right up to the applications used to acquire and analyze the data. This paper assumes no (or very little) previous experience with networks, and looks at how a networked based approach affects the elements of an FTI network, what the requirements for a basic FTI network are, and some of the things a flight test instrumentation engineer must look out for when designing or working with an Ethernet based FTI network.

Although there is much talk of networked FTI, the number of actual flying systems is limited. Those that exist tend to be large - Boeing 787 [1] and MMA, Airbus A380 [2] and A330 MRTT [3] are examples, with half a dozen other smaller systems flown. It would be a mistake to assume from this that networked FTI means large installations. A fully functional FTI network with just five elements will be described later in this paper.

As yet, there are few standards adopted in this area. The CTEIP iNET initiative is attempting to fill this void with various working groups tasked with producing standards on all aspects of networking[4], and there are various published papers on specific elements (for example the Airbus IENA packet standard[5] and the IEEE-1588 time synchronization protocol[6].

The rest of this paper will address what is involved in networked FTI and the areas the standard needs to address to ensure we can evolve smoothly to a fully networked approach.

## NETWORKED DATA ACQUISITION

### A Change of Paradigm

Before getting into the details of a minimal networked acquisition system it is worth looking at a couple of important conceptual difference between traditional FTI networks and Ethernet based networks.

In a traditional network with more than one data acquisition unit (DAU), typically one DAU was designated the master and the rest were slaves. The master was responsible for synchronizing the slaves, and gathering data from them for transmission as PCM. A networked FTI architecture replaces tyranny with democracy – there is no specific master, just a collection of peer nodes, each transferring data into the network. The only exception to this is with respect to time – one of the nodes is the "time-master" and is responsible for ensuring all the other nodes have the same notion of the time of day.

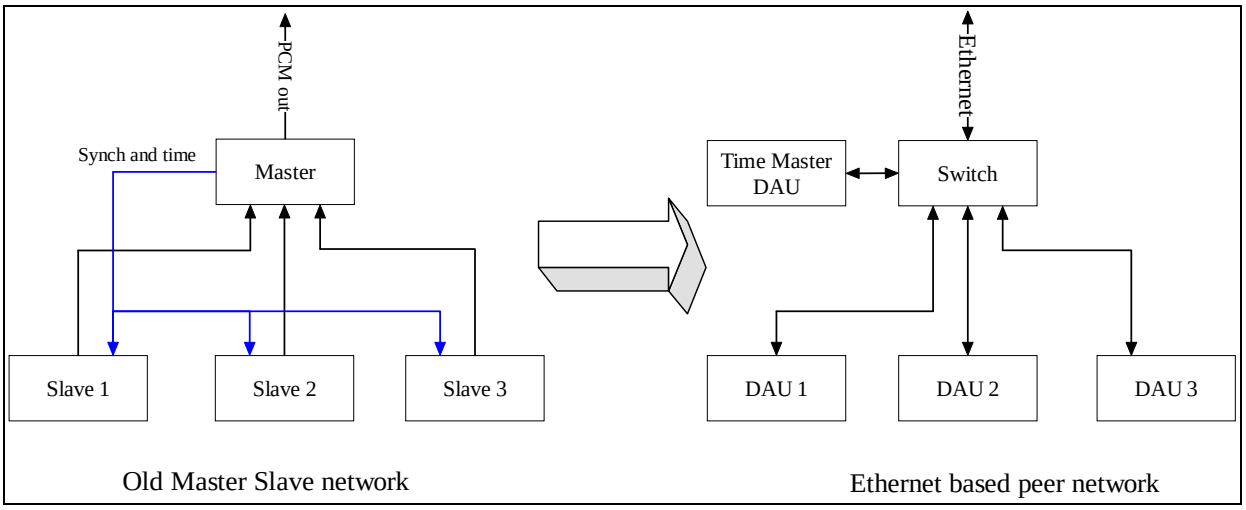


**Figure 1: Old master slave system as implemented in a network**

Another important change is that PCM systems of old were generally deterministic. In other words, it was possible to know in advance exactly how long it would take for a sampled parameter to reach its destination. This meant that in the older, now obsolete, multi-plexed systems it was possible to tell the sampling instant of a parameter by its location in the PCM frame, once you knew the time-stamp of the frame itself. Modern synchronized systems would guarantee that all data in a PCM major frame was sampled in the same epoch (coherent), so all you needed to know was the time stamp of the frame and you could calculate the sampling instant of each parameter without reference to the frame layout.

Ethernet IP networks are non-deterministic. You can time-stamp the data to know when it was sampled, but you do not know for sure when it will get to its destination. This presents special

problems when dealing with real-time latency budgets, or when trying to stuff networked data into a deterministic legacy PCM stream (see section X below).

Finally, there is a conflict in operational requirement between what is expected of a traditional system versus a networked system. In the traditional world we are used to devices becoming "live" very quickly (< 10s) and spitting out data without being told to. Brown outs or power failures should have a minimal effect on our data acquisition. However, in a networked world, users are more used to the "network" taking time to configure itself and being non-operational for a couple of minutes. This is one element of networks that should be avoided in an FTI application.

So, with these important features in mind, let us look at the basic elements of a networked FTI system.

## NETWORK COMPONENTS

Figure 2 shows the minimal configuration of a networked FTI system. (Note – the trivial case of a single DAU is not discussed, but of course there is no reason why a single DAU cannot transfer its data over Ethernet!) The core elements are: 2 or more DAUs, a switch, and a time master (called as Grandmaster in the IEEE-1588 terminology). Extra elements like an IP recorder and a legacy PCM output will depend on the application but are usually present. This basic system can be expanded by adding extra DAUs and switches, almost without limit, to meet the needs of the application.

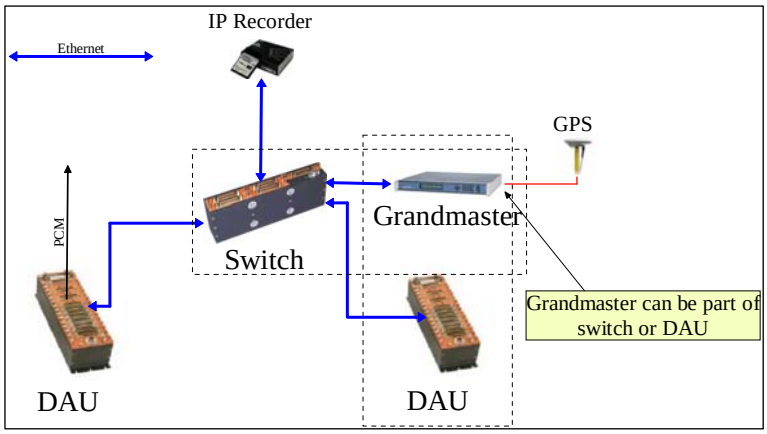


**Figure 2: Minimal Network Configuration**

Note that this picture assumes that time synchronization is handled by IEEE-1588 (or Precision Time Protocol, PTP). Any alternative approach requires more wiring to the nodes to perform the synchronization and/or time transfer so is sub-optimal from a network point of view. However, other standards (e.g. IRIG-B) can be used with additional wiring, removing the need for a grandmaster.

The Ethernet standards support 10/100/1000Mbit copper and fiber cabling. Most DAUs support 100MBit/s network interface. A 1GB interface is not necessary unless the DAU itself has a very high throughput. However, 1GB may be necessary as a network backbone for feeding a recorder as this will carry an aggregate of data from many DAUs.

### The Data Acquisition Unit

The data acquisition unit must have:

- the ability to synchronize to an external time source using Precision Time Protocol.
- the ability to gather and time-stamp data

- the ability to transmit the data over Internet Protocol (IP) continuously and without instruction
- a very short boot-up cycle (< 10s)
- the ability to have its address on the network (the IP address) set permanently to avoid having to wait to be told what it is during a power recovery scenario

If the DAU has the following features it makes life a little easier when configuring and debugging:

- the ability to report its hardware address (called the MAC Address) on request (this is done via a protocol called Address Resolution Protocol, or ARP)
- the ability to respond to some standard configuration and status requests (this is usually supported via a protocol called Simple Network Management Protocol – SNMP).

The KAM-500 from ACRA CONTROL is a standard flight test data acquisition unit which, with the substitution of the standard PCM controller, becomes a networked DAU[7] that exceeds these minimal requirements.

**The Switch**

Because our DAUs are all peers now, we need some way of gathering the data from the disparate sources and sending it to where it should go. In networks, this role is performed by a hub, a switch or a router. These terms (and others) all refer to devices that take Ethernet traffic in one (or more) bi-directional ports and sends it out one (or more) ports according to source and destination information contained in the traffic itself. There are various categories of device that can be used here, each of differing levels of complexity. At the simplest, a hub takes any data that arrives at any port and copies it to all the other ports. It has no intelligence, is simple to use, but very inefficient in terms of traffic management. At the other end of the scale, a managed switch can route traffic with different priorities and to different places all based on the content of the headers of the IP packets.

With two or more DAUs, the FTI network will need at least one switch. This switch needs the following minimum set of requirements:

- The ability to handle full-bandwidth traffic on all its ports simultaneously, In other words, so long as the aggregate throughput of all ports does not exceed the bandwidth of the network, no packets should be lost.
- PTP support (either as a PTP switch or in transparency mode [8])
- Bounded switching delay. There are no devices that will give a deterministic delay as the packet passes through so the best we can hope for is a known maximum delay.
- Multicast (single source, multiple destinations) support.
- Up and running in a short time (<10s)

The functionality required, beyond this basic level, will depend on the application and the size of the network. However, useful features include:

- network status reporting via SNMP (throughput, packet loss etc.)
- traffic management and prioritization

An example of a simple, small, rugged switch designed for extreme environments is the NET/SWI/001 from ACRA CONTROL. This switch is designed to be up and running in a very short time (<10s), will operate over an extended temperature range (-50C to +105C) and exceeds MIL-STD-810 environmental specifications. It is intended to be used in small FTI environments where a larger, fully managed switch, cannot survive.

**The Grandmaster**

Somewhere, the network has to get a notion of what the time is. In a distributed data acquisition system each node maintains its own copy of time. So the problem then becomes one of both distributing the time to all nodes, and subsequently ensuring that they remain time synchronized.

In networks the best solution for this is PTP[9]. With PTP a so called “grand master” distributes time to all nodes at regular intervals. The nodes send check messages to determine network transfer delays and compensate accordingly. With such techniques, nodes can be kept in sync to better than 100ns. Once synchronized, nodes can use various techniques to guarantee sampling synchronicity.

Although there are several grandmaster manufacturers, at the time of writing no commercial stand-alone grandmasters are ruggedized for flight. There are ruggedized switch/grandmaster combinations available, and some DAUs (e.g. the KAM-500) can be converted to a grandmaster by the addition of a GPS time card. A combined solution like this saves another box on the test article.

**The Recorder**

An IP recorder captures all network traffic. There are various types available. One approach simply records the electrical levels on the bus and then allows the messages to be replayed later. Another approach understands the messages well enough to unpack the data and store it. The latter approach allows read-while-write data recovery during a test, permitting older data to be randomly accessed and then relayed when requested. An example of this type of ruggedized IP recorder is the KAM/MEM/005 expansion module for the KAM-500.

## NETWORK MANAGEMENT

Standard Ethernet protocols provide options for managing the elements of a network. A popular standard is SNMP[10] which defines a simple messaging protocol (GET, SET, etc.) for communicating with network devices. The set of variables that can be managed is defined by an application specific document called a Management Information Base (MIB).

The advantage of SNMP is that it is simple, fairly easy to define and implement, and powerful. The disadvantage is that it requires, by definition, a network manager. Something has to originate the GET and SET commands and capture and collate the responses. This is typically a computer. While this is feasible in a large flying installations, it is usually impossible for smaller installations. (SNMP support may still be useful in a smaller installation for pre or post flight configuration, checking and debugging).

So for in-flight network management the system should support the old fashioned PCM approach of embedding status and operational information in data words that can be captured or transmitted with the data. For most applications SNMP is a “nice to have” but not essential.

## DATA TRANSFER

When it comes to data transfer the two most popular protocols used are Transmission Control Protocol (TCP) and User Datagram Protocol (UDP). TCP guarantees delivery, but carries a lot of overhead and has unbounded latency. UDP is a lightweight “fire and forget” protocol that does not guarantee delivery (rather like PCM) and has a bounded latency (for messages that are not lost!).

In practice, UDP is the preferred protocol for streaming data for the following reasons:

- In streaming data it is often better to lose data than for it to arrive too late. Latency is more important than reliability
- The overhead of acknowledgement and resend associated with TCP is too expensive
- We can design the network a priori to minimize the probability of packet loss. (Real FTI networks have seen packet loss probabilities better than 10-9).

Real Time Protocol (RTP) is a protocol built on top of UDP that has some further elements to improve reliability and delivery that may have applications in flight test although at the time of writing there are no implementations available[11].

An important assumption about FTI networks that marks them apart from a normal office network is that they can be designed to be “lossless”. This is because we know, in advance, just how much traffic each node can generate based on its data inputs (both synchronous and asynchronous) and the sampling rate. We can then ensure that the network backbone has enough capacity to carry all the possible traffic. The addition of good quality switching infrastructure with full buffering then ensures that traffic will not be lost at peak times (although the transfer delay through the network may well vary with network load). This is one of the reasons that UDP is attractive. (Indeed, TCP works against us here as it resends lost packets, meaning that we cannot size the total traffic in the network in advance because we cannot predict how many re-sent packets will be carried at any time).

### DATA CAPTURE AND PROCESSING

Another important implication of network FTI is that the data processing system needs to be able to process and interpret the Ethernet data. The hardware configuration is much simpler, of course, since a standard Ethernet port is all that is required. At present, there is no packet format equivalent to IRIG-106 Chapter 4 for PCM. However, all packet formats have several things in common:

- They contain a time stamp. This is the time that the first data sample in the packet was captured (not the time the packet was transmitted or received – neither of these times have any relevance)
- There is a sequence number. In Ethernet there is no guarantee that data arrives, or if it does, that data from multiple Ethernet sources arrives in transmission order. The sequence number greatly simplifies the processing of these packets.

The KAM-500 can generate several packet formats but the default is the IENA standard published by Airbus [5]. There are several software packages that can capture and interpret these packets natively, for example IADS® from Symvionics.

## PRACTICAL APPLICATION AND EXPERIMENTS

ACRA CONTROL has been working with FTI networks since 2004, with the first flight of networked data equipment occurring on the Airbus A380 in 2005. Since then the hardware and software has been refined based on our experiences and we have installed and experimented with several different types of Ethernet based data acquisition systems.

While the technology is still evolving the outcome has been extremely encouraging. Some of the issues encountered, and the possible solutions are described in this section:

### Networks are slow to "boot up"

It takes time for the network infrastructure to learn about itself, where nodes are, what traffic goes where and so on. This might be acceptable for certain applications, but there are many places where power drop-outs are common and fast recovery is essential. Steps to minimize this are:

- Ensure that any DAUs in the system boot-up quickly (<10s). 30 seconds or more is too long.
- Do not use dynamic address configuration (DHCP). Set the addresses of the nodes manually so that they are addressable the moment they are live.
- Use simple unmanaged switches (if the network and traffic loads are small enough to permit it) or use manual routing tables so that the switches do not have to "learn" too much
- In high traffic situations use intelligent routing protocols (e.g. IGMP) to ensure that traffic is routed only to where it should go. (This is slightly in contradiction with the previous rule of thumb, so some compromise is necessary)
- DAUs should come up "live" and transmitting, and not depend on some configuration or commands to operate. This implies that the DAU must be configured in advance and retain its configuration in EEPROM.
- Do not use a network system manager on the test article. It becomes a single point of failure, as well as the main source of "boot up" delay.

### Ethernet is non-deterministic and does not guarantee delivery

This issue is a fact of Ethernet life. The only solution is to design the network infrastructure to have more capacity than the expected traffic during the test. Assuming no buffer overflows and UDP traffic then it is not difficult to keep the delay through the network from sensor to recorder to less than 100ms. In experiments that ACRA CONTROL have performed, the delay on packets transferred over the public internet, trans-continental, has been in the region of 160ms (with a standard deviation of ~60ms).

Delivery is "best effort" and not guaranteed – but experiments have shown that in a properly designed network the probability of packet loss is negligible.

The non-determinism has an impact on the “legacy PCM problem” as described in below.

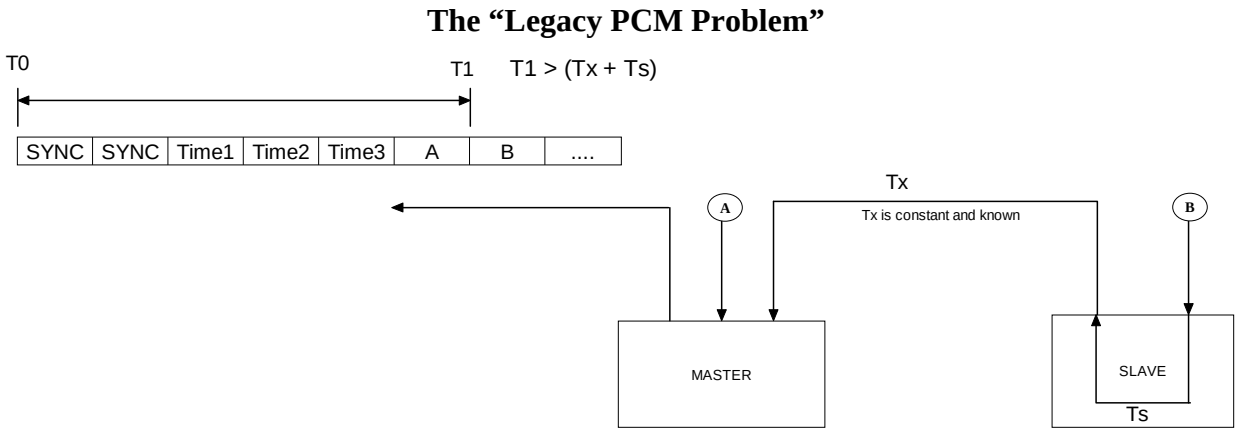


**Figure 3: Transferring data over traditional Master/Slave link**

Figure 3 shows a traditional PCM master/slave system. Data can be transferred from the slave to the master in time $T_x$ (using PCM, or CAIS or some proprietary means). Since data is sampled simultaneously, it is available for transfer $T_s$ times after the sample instance (where $T_s$ is the time to process the sample). $T_x$ is known, and is constant as is $T_s$. So, any slave data placed in a PCM frame at least ($T_x$ + $T_s$) from the start of the frame can be transferred on time and will be coherent with all the other data in the frame. (i.e. sampled in the same epoch as the frame). Typically Tx is of the order of 3 or 4 microseconds.

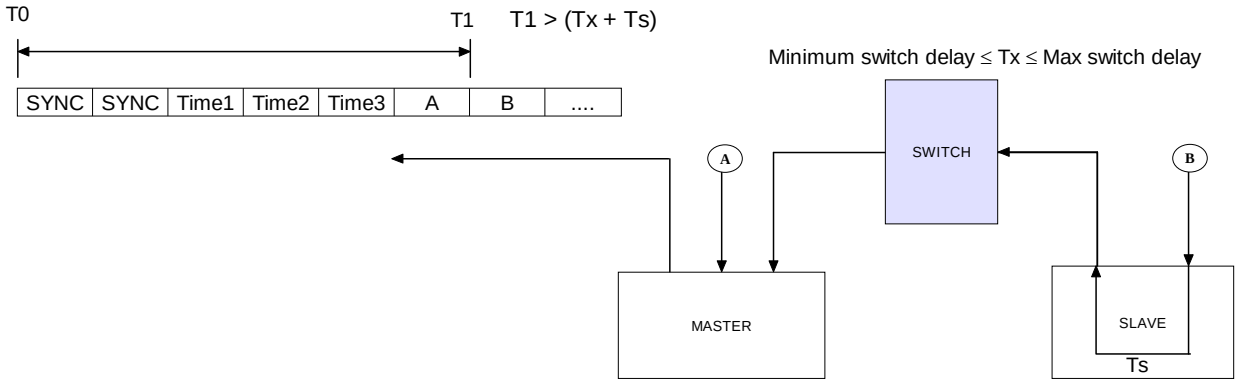


**Figure 4: Transferring data over a networked system**

In Figure 4 the same scenario is shown in a networked architecture. In this case $T_x$ is not known, nor is it constant. Switching delays tend to much longer than traditional transfer delays – 10s of microseconds. It depends on what other traffic is on the network and other factors. We can define an upper bound for it if we know the maximum switching delay for the switch, but the bottom line is that we cannot guarantee that a sample taken late in the acquisition cycle can be transferred to the PCM stream before the end of the major frame of that cycle. We cannot guarantee coherency under all circumstances. This is an unavoidable issue when we are feeding a deterministic source (the PCM encoder) over a non-deterministic medium (the Ethernet).

The solution to this issue is to treat data originating in chassis other than the PCM source chassis as asynchronous bus data. So the data is captured on the PCM chassis, time stamped and retransmitted. This is easiest to understand if the PCM frame is divided into “packet slots” that group all the data from a given remote DAU. There is no requirement to treat the data this way, the captured data could just as easily be split up and placed according to traditional PCM rules, however it is easier to manage the time co-relation if each packet from each remote DAU is captured as a continuous entity, time stamped, and placed in the frame.

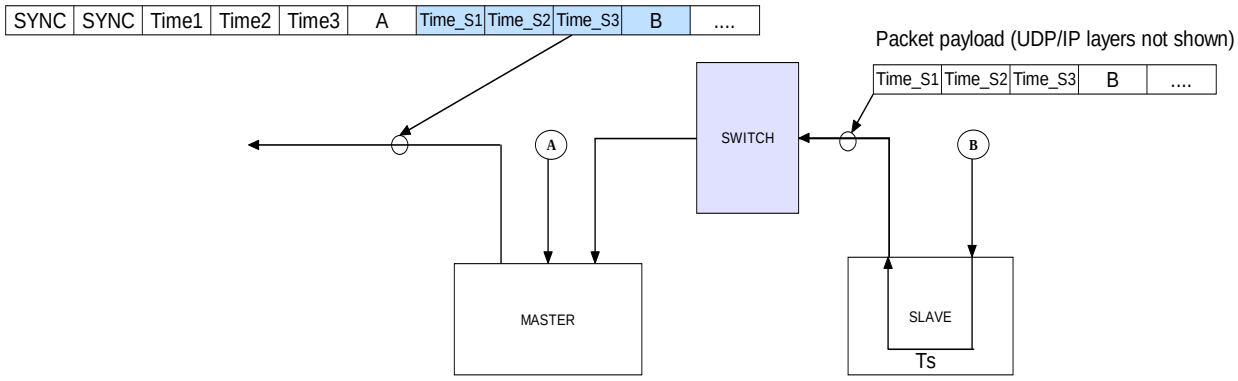


**Figure 5: Embedding packet data in PCM**

The data in the PCM frame is no longer coherent, but since each packet is coherent, and time stamped, it is relatively easy to process and co-relate the data. Note that the data is not truly asynchronous – if the DAUs are sampling simultaneously all samples will be taken at the same time so the time stamp is needed only to re-align the sample temporally.

## CONCLUSION

Networked FTI solutions are here and operating today. Nevertheless, the technology for integrating these established COTS techniques is evolving. A gaping hole is the lack of standards – something that iNET is trying to address. For an industry which is used to operating in a PCM focused world, the switch to networked FTI involves some fundamental changes in how to think about data acquisition. However, in general the use of open networks like Ethernet simplifies the way things are done and opens new opportunities.

Designing, purchasing and installing FTI networks requires some understanding of networking issues. However the DAUs that are used today are not radically different from their network enabled cousins, and with some careful choices it is easy to transfer to a network based FTI installation. The key things to remember are:

- The acquisition hardware must still survive in the hostile world of flight test: fast boot up, pre configured to operate, must survive the environmental conditions.
- Any network needs a switch and a grandmaster.
- The processing software needs to handle networked information natively.
- If you need legacy PCM, then you need to think about how to handle remote data. It can be managed, but the processing software needs to be aware of the time implications

[1] Eccles, Lee, "Network Based data Acquisition", Proceedings of European Telemetery Conference, Munich, Germany, 2008
[2] Revaux, Nathalie and Abadie, Frédéric, "A380 Flight Test Architecture: Switched IENA from Pulse Code Modulation to Ethernet LAN", Proceedings of European Test and Telemetry Conference, Toulouse, France, 2003
[3] Doyle, David and Canizares, Ruben, "Review of a successful distributed networked and modular FTI implementation", to be published in Proceedings of Society of Flight Test Engineers Symposium 39, 2008
[4] Grace, Thomas and Hodack, David, "Vehicle Network Technology Demonstration", Proceedings of International Telemetry Conference, 2007.
[5] Caturla, J. P., "IENA Packet Format and Generic Control Tools", Proceedings of European Test and Telemetry Conference Toulouse, France, 2003
[6] IEEE Standards Committee, "Precision clock synchronization protocol for networked measurement and control systems", IEEE Std. 1588, 2004
[7] Corry, Diarmuid, "System Wide Synchronization in Distributed Data Acquisition Networks", Proceedings European Telemetry Conference, Munich, Germany, 2008
[8] Nylund, Sven and Holmeide, Øyvind, "IEEE1588 Switch Transparency", Proceedings of NIST Conference on IEEE1588, Gaithersburg, Maryland, 27-29 Sept, 2004
[9] Eidson, John and Lee, Kang, "Sharing a common sense of time", IEEE Instrumentation and Measurement Magazine, March 2003
**[10]** IETF, RFC 3411, "An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks", Dec 2002
**[11]** Cranley, Nikki, "Real-time Transport Protocols for Telemetry Data and Signalling", Proceedings of International Telemetry Conference, 2008