# REAL-TIME ISSUES IN DISTRIBUTED DATA BASES FOR REAL-TIME CONTROL

M. G. Rodd

*Institute for Industrial Information Technology, University of Wales, Swansea, UK*

Abstract: This paper takes a pragmatic application-oriented view of distributed data bases to be used in Real-Time Process and Manufacturing Control. It acknowledges the strides that have been made in the development of distributed data base techniques in general, but is concerned with the problem of mapping these techniques into the Real-Time Control world. The paper takes the view that distributing data in a control system is not only an essential feature of integrated systems, but indeed is the key to their success or otherwise. However, this distribution of data has characteristics which differentiate it from other areas in which distributed data bases are used - such as air line bookings, etc. The paper discusses the essential nature of these differences and goes on to discuss how distributed data bases have to fit in with the overall control strategy, which in turn calls heavily on the use of integrated communication systems. These latter systems are themselves often based on rapidly emerging standardised systems (primarily for economic reasons), and are increasingly OSI-based. The paper concludes that to be of value, a distributed data base in a real-time control system must be designed from the viewpoint of the user rather than from that of the data base designer!

Keywords: Real-Time Distributed Data Bases, Real-Time Issues in Data Bases, Distributed Data Bases, Distributed Real-Time Data Bases, Integrated Data Bases.

## 1 TOWARDS INTEGRATED CONTROL SYSTEMS

In all areas of automation, there is a very natural, evolutionary trend towards total integration of the controlling structures. The reasons for this are very simple: economics! Clearly, with the competition which has grown up in the last 20 years to produce products which are cheaper and more reliable, and simply satisfy the market need, has required manufacturers of any product to look towards achieving total efficiency, be it in a chemical process or a discrete manufacturing plant. The key to such efficiency is to have total control. Fundamental to total control is data.

Data is now seen as a valuable resource. Just as a local control system cannot operate without knowledge of the process it is controlling, or without a set point provided from a superior system, in a similar way the managing director of a company cannot make correct policy decisions without knowing what is happening on his plant at any one time, as well as what the market needs are.

Automation is naturally not the panacea to these problems; indeed, a small, tight, manufacturing plant, run by a manager who is totally in control of that plant, and has reliable knowledge of the market demands, can be extremely efficient. However, as our plants get more complex in terms of the interactions between their various components, and indeed, as the relationships between the plants and the suppliers get more and more complicated, the only solution is to turn towards mechanical and electronic aids to achieve the integration. In non-philosophical terms, integrated computer control is simply a means by which a complex system, with high-speed responses at all stages, can be effectively tied together.

As a result, distributed computer control systems are neither magic nor anything more innovative than a natural solution to a very tough problem - a problem essentially of coping with complexity. We have learnt only too well the problems which arise in producing complex systems. Not only do we have difficulty in conceptualising how they should be structured but, more important, we have come to realise our own fallibility. We simply cannot manufacture either hardware or software with the required degree of reliability. As a result, we learn very quickly that the only way of tackling this problem is that of "divide and conquer". We are rapidly learning to divide up complex software tasks into relatively small chunks - chunks which we can understand and monitor, and at least attempt to test.

As we move, then, towards distributed computer control structures, it has become evident that we must produce some form of hierarchy in order to structure these systems. We have to harness geographically related processes and wherever possible reflect the natural integration which results. We also find that as we develop an appropriate form of hierarchy in our distributed computer control systems, the various levels of automation define the information requirements. In practice, the so-called five-layer model, as illustrated in Figure 1, seems to become relevant, reflecting as it does most practical situations.

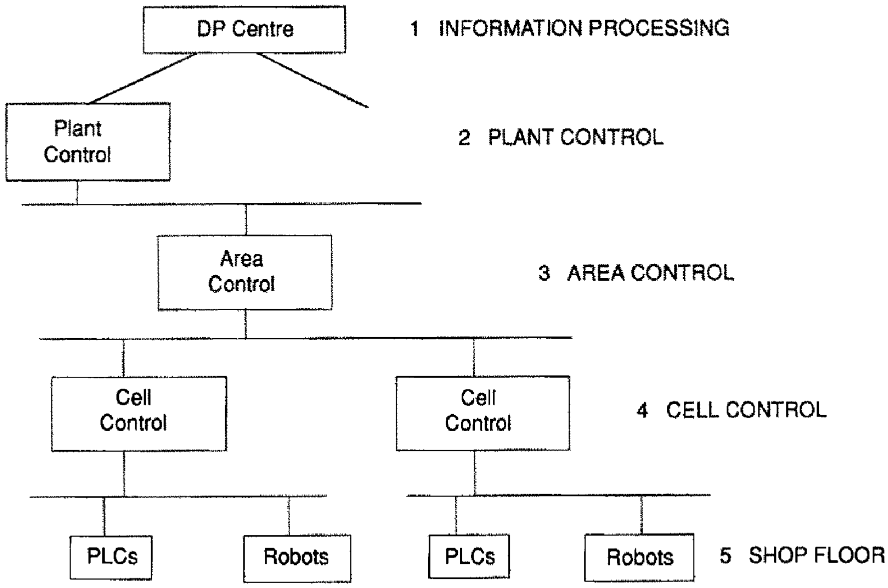


Figure 1

A further requirement of the distribution is to cope with the problem of system failure. During the early days of computerisation we learned, to our cost, the disasters which result from the "eggs in one basket" syndrome. The loss of an essential computer could be devastating. We have learned that the only way of tackling this is to distribute the control, so that the failure of any particular piece of software or hardware will lead to a situation which we can at least contain.

Naturally, all that has been discussed so far is very evident and the points do not need labouring. However, it is important to realise that fundamental to such integrated systems and their hierarchical implementation is the data which is the "lifeblood" of any distributed system. Equally important is the fact that the "veins" which carry this lifeblood are the communication channels which we instal between our computing nodes. As we will see later, the nature of these "veins" has a tremendous influence on how we move information around the plant or, in a grander way, between plants using various public services.

Turning to the question of data, our colleagues in the data processing industry will say that there is no problem, and that most of the difficulties were resolved several years ago. They point, very rightly, to the successful airline booking systems, or to the international banking structures, all of which naturally demand distributed data bases. However, when one starts mapping these techniques into our real-time control system applications, various things appear to go wrong, and there does appear to be a fundamental difference between the very successful applications of distributed data bases in the data processing field and those which are encountered on the shop floor.

## 2 REAL-TIME VERSUS ON-LINE

The thesis of this paper is that to design a real-time control system one must appreciate the real-time nature of the process that one is controlling. In essence the problem is one of distinguishing between an "on-line" computing system and a "real-time" computing system. This distinction has become extremely blurred over many years, and it is critical that we return to some fundamental understanding.

Essentially, in a real-time computing system the correctness of the system depends not only on the logical results of the computation but also on the time at which the results are produced. "Time-critical" or "real-time" does not necessarily imply "fast", but does mean "fast enough for the chosen application".

In essence, in time-dependent processes, data is judged not only on its actual value but also on the time at which the value was obtained. For example, a sensor measuring an attribute of a dynamic system should specify not only the actual value of that attribute but also when that parameter was sampled, since of course the system is time-varying. Real-time data in any distributed network is thus valid only for a given period of time and indeed has meaning only if the time of creation is specified. This forces designers increasingly towards the concept of atomic units of data in which the value and the time of creation are inseparable. Fundamental to all this is that we are completing control loops. The designer of a simple PID controller assumes that the variables which we are measuring are obtained when we think they are. However, if we put a transmission system between the point of measurement and the point at which the measurement is utilised, that transmission system has its own characteristics. These characteristics must be considered in solving the control algorithm problem. There is no point in comparing the value of a temperature sensor with a required set point unless we know at what point in time we are actually measuring that temperature. On the other hand, in the on-line situation, we are merely required to be able to have discussions directly with our computer. If that computer takes a little bit longer to handle, say, a request, then this is not serious, except for maybe causing some irritation.

In the real-time system, any delay could cause the plant to be in serious error.

It is clear, therefore, that the data which flows around our distributed computer control system will, in many cases, be time-dependent and not merely value-dependent. It is important, therefore, that we look at the nature of the data in our real-time control system when we start developing methods of transmitting and storing that data as the links in integrated computing systems, i.e. the sharing of data.

Professor Hermann Kopetz of the Technical University, Vienna, proposed some years ago in a public lecture that the characteristics shown in Figure 2 give us a good clue as to the difference between "real-time" and "on-line". He and many other workers in the area have suggested that there are various degrees between these two extremes. However, it is felt in this paper that this distinction is not really necessary and that a simpler and more reasonable approach would be to define a real-time process as one in which the temporal characteristics are absolutely defined, whereas an on-line situation is one in which these can be degraded without any serious loss of performance.

As a result, it is suggested that in real-time control, data has meaning only when it is associated with time. This implies, in the first place, that all data must be time-stamped at the time of creation in order to provide that real-time information. Of course, the direct consequence of this is that all nodes in a system must have access to a globally agreed real-time clock. Again, work by many authors has pointed to ways in which such clocks can be created. In essence they require a good-quality local clock at each computing node, with some form of synchronisation by means of frequently transmitted time messages, or by a separate timing channel, or, finally, by access to some internationally transmitted standard.

However, merely time-stamping the data does not solve all the problems. A particularly important issue which must be considered, and which will be referred to later, is that of consistency. Not only must data be consistent in value, but this consistency must be referred back to time. Two aspects of this are well illustrated by the following examples.

In the first place, consider the implementation of redundant sensors. Say, for example, one is measuring a single temperature variable but using three redundant temperature probes, with each probe producing its version of the temperature. A further controller must make use of this temperature information, and the question of course arises as to which temperature value to use. Whilst in the past this would probably have been handled by a voting mechanism, clearly if each probe transmits its information separately over the communication network, then there is no way for the controlling node to know how to implement this voting, since it will not know when each value has been created. The only way to cope with the problem is by time-stamping those temperature variables.

The second example is a very complex one, relating to the problem of an operator getting a consistent picture of an overall plant. Consider that the operator might be at least two or three networks away from the actual plant; then information which is presented to him on the screen will have to travel up the communication networks before it can be displayed. Information which is obtained locally could well get to him before information which is obtained from a remote site. If, however, he (or an accompanying artificial intelligence system) is required to base a decision upon the data presented, this is an impossibility unless the actual time at which all the data was created is available to him, thus ensuring that consistency can be checked.

Both these points have great bearing on the structure of the distributed data base, and this will be referred to later in this paper.

## REAL-TIME versus ON-LINE

| CHARACTERISTIC | HARD REAL-TIME | ON-LINE |
|---|---|---|
| Response time | Hard | Soft |
| Pacing | By Plant | By Computer |
| Peak Performance | Must be Predictable | Degradable |
| Time Granularity | Less than 1 mSec | About 1 Sec |
| Clustering | Important | Not Essential |
| Data Files | Small-to-Medium | Large |
| Data Integrity | Short Term | Long Term |
| Safety | Critical | Non-Critical |
| Error Detection | Bounded by System | User Responsible |
| Redundancy | Active | Standby |

Figure 2

It is also relevant at this stage to point out that when one looks very closely at the hierarchical structure which is developing in a distributed computer control system, one sees that the types of data which occur at the various stages in the process vary greatly. This, again, will be referred to later in this paper.

## 3 COMMUNICATION SYSTEMS IN DCCSs

So far we have mentioned the fact that in a Real-Time Distributed Computer Control system it is critical to take into consideration time and its effect on data. Supporting our distributed computer system in the real-time control world, of course, will be communication systems, and their very nature will have a great influence on what is moved around the plant, and how. We have too often been guilty of assuming that because we appear to have very large bandwidths available to us in our networking systems, we are free to move as much data as we wish around a plant. In reality, however, once we try to move this data reliably and efficiently we suddenly discover that our communication systems start to become bottlenecks in themselves. Also, we suddenly discover that the nature of the communication systems can greatly influence the efficiency of moving the data. Particularly relevant, too, is that once we realise the real-time nature of data in process control then we have to look at the real-time characteristics of our communication systems.

Of course the fundamental issue in the development of any interconnected computer structure is the problem of incompatibility. Essentially, in a distributed computer system what we are trying to do is to permit various computers from various manufacturers to share various pieces of data. Whilst from the point of view of the supplier it would be desirable for all our computers to be from his own stable, and therefore totally compatible, we soon realise that in the distributed computer control world we are at the mercy of a variety of suppliers, each of whom has particular strongpoints; thus, certain suppliers can provide very fine data processing hardware and software, whereas others specialise in hard-nosed factory floor compatible controllers. The difficulty of producing a distributed system is to bring all these bits and pieces together. The compatibility issue revolves around hardware, software and, of course, data. Of course, we would like to be free to design our own communication systems and indeed our own programs. However, the reality is somewhat different!

In practice, solving the compatibility problem is extremely difficult and we are only beginning to make some form of impact on approaching it. The key has to be standardisation - whether or not we or the suppliers will acknowledge it. Naturally, at the upper data communication level much progress has been achieved of late in the move towards OSI-based protocols, and in other fields, too, the success of standardisation is evident - for example Ethernet and TCP-IP are good examples of standards which have met their desired marks.

Of course in the hard world of process control standardisation is critical, but unfortunately we are relatively small as an industry, when compared to the data communications field. Whilst we would like to set our own standards there is no way that we can economically go down this route and therefore we tend to be the "step-children" of the data communications industry.

Realising the problems of standardisation, General Motors initiated its MAP exercise some years ago and at the same time Boeing Corporation developed its TOP initiative. In both cases the idea was to produce a profile of protocols which would be appropriate for their particular areas of application - in the case of MAP, manufacturing, and in the case of TOP, technical offices. It was realised right from the start that there was no point in going it alone, and that the prime way ahead would rather be to select from an agreed and internationally acceptable protocol profile the appropriate layers, and also within those various layers to select certain protocols and, where necessary, to inject extra emphasis to direct the move towards standardisation.

Although MAP has attracted much criticism, it has undoubtedly been very significant in that it has brought to the fore the need for standardisation. The exercise has also hastened the development of protocols which are appropriate (at least to some degree) to manufacturing and to the process industry. It is very clear, however, that the products which are resulting from this exercise are in many cases too expensive and too complicated to be of value in the hard world of automation. However, at the same time it must be acknowledged that certain aspects of the MAP exercise will undoubtedly prove to be extremely important in the long term.

Of particular importance, for example, is the development through MAP of MMS - Manufacturing Messaging Services. MMS provides a standardised manner of sending messages around a process or manufacturing plant. The standard itself defines a core of services, and then it is left to specialised groups to provide so-called "companion standards", which expand on these services for particular application areas. Indeed, it is probably fair to say that the standardisation exercise should have started at the MMS level, leaving many of the lower-layer and somewhat technical issues for later, in that many of the lower layers, such as the physical layer, will be highly influenced by technical achievements and current issues (for example, whether it should be Ethernet or Token-passing) might well eventually be rendered redundant.

The point is that there can be little doubt that the standardisation exercises are extremely critical to the design of future distributed computer control systems. They will also have serious impacts on how we handle data within DCCSs. We cannot ignore them: they are economically critical and simply to throw out the MAP exercise would be extremely naive. However, to look at the MAP exercise, to take what is good out of it and then to assist in the development or redevelopment of some of the layers which are proving to be troublesome, makes eminent sense.

Another aspect of the trend towards standardisation fits in extremely closely with the idea of development hierarchies within the integrated control system. Exercises such as MAP have acknowledged that we will ultimately see, say, a high-capacity data backbone running through the plant, fed by bridges or gateways from less-capable, cheaper networks, which in turn could well be fed by very low-cost field-bus type networks. After much debate and academic discussion has taken place as to the ideal structure there does appear to be an increasing consensus that our systems will ultimately move, in the case of real-time Control Systems, to a structure like that illustrated in Figure 3.

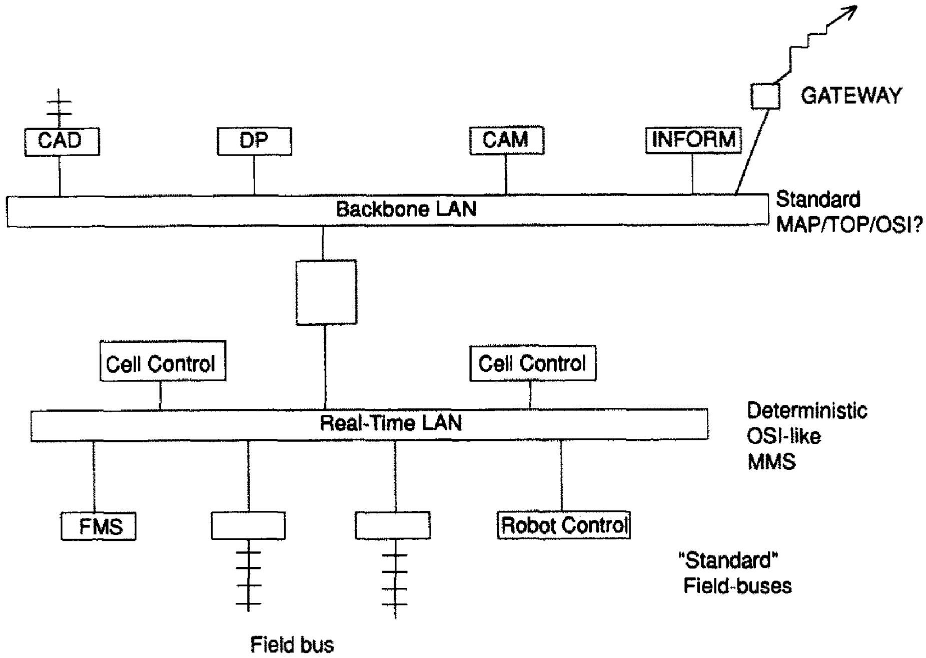


Figure 3

In this figure one sees, again, the idea of having a powerful backbone LAN which can in turn talk via gateways to wide area networks. The backbone will also communicate into more deterministic high-speed networks which we have chosen to call real-time LANs. Whilst it is clear that current OSI proposals do not cope with the requirements of such real-time systems, there is no doubt that in the future they will. It is interesting to note also that many important companies operating in this area are envisaging models very similar to that proposed in Figure 3. It is critical also to note that below the real-time LAN level there will be a move towards so-called Standard Field-Buses and here the current international attempts to standardise the field-bus are particularly relevant.

Important throughout this is that whilst various components are not yet defined through International Standards, there is no doubt that in time they will be. More important, though, is the acceptance of the OSI model; probably even more important, is the acceptance of some of the higher-level services such as MMS, FTAM, Directory Services, etc.

## 4 LEVELS AND FUNCTIONS

Studying Figure 3 in detail and looking at the data requirements at various levels one begins to see a pattern emerging.

At the higher levels, such as the backbone LAN, one sees that the functions which are typically required include

- File transfer
- Electronic mail
- Remote file editing
- Status reporting to operators in higher levels of management
- Data acquisition from lower levels
- Control of lower levels
- Program transfer

In essence, at this level, one is moving a large bulk of data around, typically between relatively large machines. This could be between CAD systems, or by sending plant status information through to management computers running production scheduling or manufacturing materials requirement planning. At this level one can see the need for efficient, but safe, transfer of bulk data. However, if one is additionally utilising this information for decision-making, then it is necessary also to be able to send time-tagged information across this system, otherwise no useful decisions can be made! However, one can accept at this level of networking that delays will occur, particularly when large files are being transferred. It is clear also that one could almost consider this an on-line environment, provided that there is some form of access to real-time data where necessary. Moving to the lower levels, particularly at the real-time LAN level, and in fact at the field-bus level, one sees a slightly different collection of functions. Here one is really getting into the real-time control world, and the following characteristics start to emerge

- Minimal defined message delay times
- Deterministic behaviour - particularly under crisis situations
- Real-time synchronisation
- Assured data consistency
- Inherent message redundancy
- State broadcasting
- Regular and frequent data updating

Essentially, we are talking here about highly efficient data transmission. Speed is important, but only to the extent that it must match the response of the plant. Thus a very slow plant will not require very high transmission rates. However, what it will require are deterministic rates. Information which flows at this level is typically relatively small in quantity, but critical in terms of its determinism. For instance, if a program does not complete on time, this could lead to serious error conditions. It is important also to realise at this stage that the network system must work most efficiently under the worst possible conditions.

Looking at our networking (or, in fact, internetworking) strategies in this way, gives us a clue as to how they may best be implemented, and what technologies are appropriate at which level. That there are different functions cannot be doubted, but a fundamental criterion has to be that where data is associated with time then the communication systems must inherently support this.

It is important also to emphasise that we cannot ignore international standardisation. As much as we would like to reject many of the proposed standards, for economic reasons we simply cannot. What we have to do is work with the standards, and alongside them, in striving towards systems which do fully meet our real-time requirements.

## 5 DATA BASES IN DISTRIBUTED COMPUTER CONTROL SYSTEMS

Finally, we look now at the problem of distributing our data bases in a real-time control system. Again, we must point out that the viewpoint taken in this paper is that the real-time control system starts at the highest level and works all the way through, right down to the plant regime. Distributing our data is, therefore, a very natural function, not merely to ensure security but to assist in meeting our time objectives. It is extremely important to examine why we move data about a plant and this could well be the key to the design of the appropriate data base structure.

The point is that at the lowest level we might just be measuring a simple temperature variable and using this to control the output of a temperature controller. At the higher level though, this information is not required. What is required is the knowledge that the controller is functioning and keeping its temperature to a specific value.

This analogy can be carried on throughout the process. The Managing Director of a company might merely want a simple spread sheet which contains information which can allow him to make appropriate decisions. He doesn't give a damn about the value of the temperature setting! What he does give a damn about, however, is the fact that the process which uses the sensor is operating, and what the actual production rate is. He needs a complete picture of what is happening on the plant, but a highly refined one.

Thus, the refining of data becomes extremely important. The simple principle should always be used - don't transmit anything unless it is needed at the other end! Thus, whilst we have regarded our communication systems as having virtually limitless band-width, in fact this is far from the truth and we are learning at our cost that once we provide effective and reliable protocols, they impose such loads on our communication systems that a 10Mbit per second communication system suddenly has an effective band-width of a handful of messages per second!

The guideline, therefore, to the distribution of our data bases has to be that we maintain the data where it is actually required. Data must be seen as a critical resource, and must be made available only where, and when, it is actually required. This in turn requires us to know, at a very early stage, who is going to use what, and when. Again, if the user requires access to data in real-time then this facility must be provided to him. Essentially, then, we are distributing the data for the convenience of the users, be they Managing Directors or local controllers.

In practice, it is found that the actual size of the data base is in itself a function of where in the hierarchy it sits. Experience is showing that the data bases at the lower levels, which might well be referred to as the real-time data bases, are relatively small in size, whereas at the higher levels where the data bases reflect the global situation, we end up with extremely large data bases. Of course it is also common experience that in many processes at least three quarters of all data gathered is collected on a Write Only basis, i.e never again used! (This is probably more a reflection on our inability to specify our requirements fully, though.)

Key, too, to the design of our data base structure is the question of redundancy. There can be no doubt that as our plants become more and more complex, we must distribute the data, just as we are distributing the hardware and software to cope with the problem. We have to make certain, therefore, that we can replicate data bases in such a way that if we lose either a data base or the associated computer, we can either bring in new hardware or execute programs on existing hardware.

This latter problem is effectively the same as is met whenever we want to copy any information to more than one data base - the problem of data consistency.

Data consistency in real-time data bases is of fundamental importance and must be taken into consideration at the earliest stages of any data base design. The problem is naturally only too well known in the distributed data base community in on-line situations, but when we start to apply these techniques in the real-time environment the situation changes. Logical techniques are no longer of any value. The only techniques which stand a chance of working are those related to real-time. Data base consistency in the real-time environment can only be ensured by taking into consideration the real-time nature of the data and the absolute binding which occurs between value and time. Therefore, solving the consistency problem can only be effectively undertaken by considering the real-time nature of the data.

Another critical characteristic of the design of real-time distributed data bases has to be the real-time nature of the process to which they are being applied. If one is looking at the lowest level where one is using the data actually to control a real-time control loop, then it is very clear that the access to the data base must itself be totally deterministic. So, just as we require determinism in the communication system which has been discussed previously, we must also strive towards totally deterministic data bases - particularly at the lower levels.

At the higher levels, whilst the access times themselves do not have to be deterministic, we have to cope with the problem of consistency. Here the point is that we are providing information to an operator, or at least to someone who is having to make a decision based on global performance of the system under their control. In order to do this we have to present them with a completely consistent picture of the plant. As we have stressed in this paper, the actual value of any particular variable has significance only if it is associated with the time at which it is created. Therefore, to provide an operator with a consistent picture, this component of the information must be considered.

## 6 CONCLUSION

This review paper has stressed the fact that in a real-time control system we cannot dissociate the value of data and its time of creation. In the design of a distributed data base we must take this as a prime fundamental consideration. We have to accept that a distributed computer control system is indeed controlling a process which has temporal characteristics. The control system, therefore (be it at the local controller level or at the larger managerial level), must take into consideration this very simple fact of life. Data base consistency is therefore fundamental and can only be achieved with an eye on the real-time nature of the data. Decisions have to be made according to a temporally consistent picture.

The paper has also mentioned the fact that moving the data around is itself critical; not only to the overall strategy but also indeed to the data base, since it will affect the temporal correctness or otherwise of the data. Therefore, in designing distributed computer control systems the fundamental nature of the communication system is itself critical. Also, in designing the data base structure, the nature of the communication system which is going to feed information into it must be taken into consideration.

The basic conclusion has to be that we need to look towards the design of real-time Distributed Data Bases from the point of view of the application, and not necessarily base our ideas around those which might have been very successful when applied in environments which are not analogous to our control world.

## 7 ACKNOWLEDGEMENT

The author wishes to acknowledge the contributions made by his colleagues at the University of Wales in Swansea, particularly Mr Ivan Izikowitz, Mr Keith Baker, Dr Farzin Deravi and Mr Guo Feng Zhao. Also he wishes to acknowledge the input of his IFAC colleagues, Professor Hermann Kopetz of the Technical University, Vienna, Dr Gregory Suski of Lawrence Livermore National Laboratories and Dr Thierry Lalive d'Epinay of Asea Brown Boveri, Europe.