International Journal of Mechanical, Industrial and Aerospace Sciences
ISSN: 2517-9950
Vol:3, No:4, 2009

Efficient real-time remote data propagation mechanism for a Component-Based approach to distributed manufacturing

V. Barot*, S. McLeod, R. Harrison, and A. A. West

Abstract**—** Manufacturing Industries face a crucial change as products and processes are required to, easily and efficiently, be reconfigurable and reusable. In order to stay competitive and flexible, situations also demand distribution of enterprises globally, which requires implementation of efficient communication strategies. A prototype system called the "Broadcaster" has been developed with an assumption that the control environment description has been engineered using the Component-based system paradigm. This prototype distributes information to a number of globally distributed partners via an adoption of the circular-based data processing mechanism. The work highlighted in this paper includes the implementation of this mechanism in the domain of the manufacturing industry.

The proposed solution enables real-time remote propagation of machine information to a number of distributed supply chain client resources such as a HMI, VRML-based 3D views and remote client instances regardless of their distribution nature and/ or their mechanisms. This approach is presented together with a set of evaluation results. Authors' main concentration surrounds the reliability and the performance metric of the adopted approach. Performance evaluation is carried out in terms of the response times taken to process the data in this domain and compared with an alternative data processing implementation such as the linear queue mechanism. Based on the evaluation results obtained, authors justify the benefits achieved from this proposed implementation and highlight any further research work that is to be carried out.

Keywords**—** Broadcaster, circular buffer, Component-based, distributed manufacturing, remote data propagation.

I. INTRODUCTION

GLOBALISATION has led the manufacturing industries to face a crucial change from a vendor's to a customer market. Since it is a customer who controls the global market, change in their demands and expectations results into a dramatic fluctuation to the entire production process. In order to stay competitive by satisfying customer demands and expectations; industries must shorten their product life cycles, reduce the time to market their products, add variations to their products, satisfy customer demands quickly and reduce their investment costs. Furthermore, these consequences imply more complex products, faster changing products and faster introduction of products [1].

Modern manufacturing production systems have to respond quickly to these continuous changes to meet the demands faced in this emerging agile manufacturing environment [2]. This requires systems to easily be reconfigurable to accommodate changes in the production process [3], [4], and geographically distribute manufacturing activities in order to stay competitive and flexible [5]. A good communication strategy has to be adopted to satisfy the requirements identified by this distribution of manufacturing activities. Manufacturing information must be delivered efficiently in soft real-time basis to a number of global supply chain engineering partners regardless of their distribution nature and / or their mechanisms. In order to achieve this, a prototype system has been developed at Loughborough University. This prototype is called *"Broadcaster"*, which has been designed as a data propagation tool. Its aim is to efficiently deliver real-time manufacturing information to a number of remote distributed partners regardless of their geographic locations or implementation mechanisms.

The main aim of this paper is to highlight the implementation of an efficient data propagation mechanism presented for the "Broadcaster" prototype. This is to distribute information from the manufacturing sources to the remote manufacturing destinations, reliably without compromising the performance metric. The description of this approach is presented together with some evaluation results which are obtained from this implementation. The work concentrates particularly on the reliability and performance of this adopted mechanism. This paper is organised as follows. The next section describes the Component-Based automation approach followed by the section three which details the data propagation mechanism together with the algorithm which has been implemented. Section four shows the overall test bed set up used to obtain the evaluation results. Conclusions are drawn in the section five of this paper, and based on the results obtained, benefits and further work is justified.

II. COMPONENT-BASED FOUNDATION

Component-Based (CB) approach not only meets the manufacturing requirements identified in the section one of this paper, but it also addresses the reconfiguration and reuse problems faced by the manufacturing industry [4]. CB paradigm aims to replace the existing widely used PLC based

The authors are with Wolfson School of Mechanical and Manufacturing Engineering, Loughborough University, Loughborough, LE11 1HF, United Kingdom. Their emails are {V.Barot, C.S.Mcleod, r.harrison, a.a.west}@lboro.ac.uk. (* is the corresponding author and can be contacted by phone +44 1509 227683)

or a PC based system architecture [6]. The fundamental concept of this approach is to develop systems from component libraries. The basic idea is that new systems are composed from components that are already developed, tested and validated. Therefore, system development requires only some of the new components to be tested while using majority of the pre-built and pre-tested components from the library. This reduces the costs and efforts for developing a system, as well as, reduces the time to market the product in order to meet new customer demands.

A simplified hierarchical version of such an approach is shown in the fig 1 [4], where "System" corresponds to any complete working machine. The "Sub System" is an autonomous section of the system containing one or more components. The CB approach provides complete control system functionality such as error monitoring, user interface system and safety circuits at the "Sub System" level. A "Component" is a physical node on the control network which encapsulates the necessary functional and physical properties, knowledge and characteristics of a manufacturing system [3]. Composition of any required system can thus be done via readily integrating various components. An "Element" corresponds to any machine input / output device such as a sensor / actuator respectively, or a more complex drive. Each element has unique "State" behavior.

An implementation of this CB approach has been described by Lee et al [3] and Harrison et al [7]. A detailed discussion of this paradigm is beyond the scope of this paper; however it is important to appreciate the major identified engineering environments of this approach to implementation. These are shown as a high-level system representation in the fig 2. The current scope of the work involves the prototype "Broadcaster", which is responsible for distributing real-time machine information efficiently to a number of subscribers (i.e. supply chain clients resources) such as a Human Machine Interface (HMI), Visualisation models / views, etc, via a client manager interface designed within the prototype.

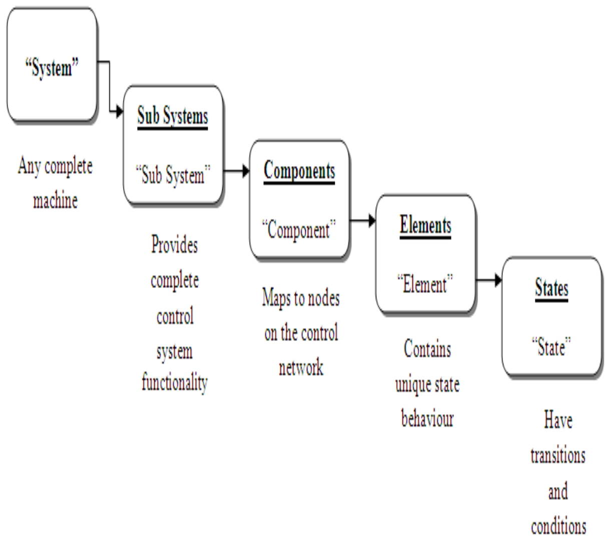
"System"
Any complete machine
Sub Systems
"Sub System"
Provides complete control system functionality
Components
"Component"
Maps to nodes on the control network
Elements
"Element"
Contains unique state behaviour
States
"State"
Have transitions and conditions

Fig. 1 CB approach hierarchy

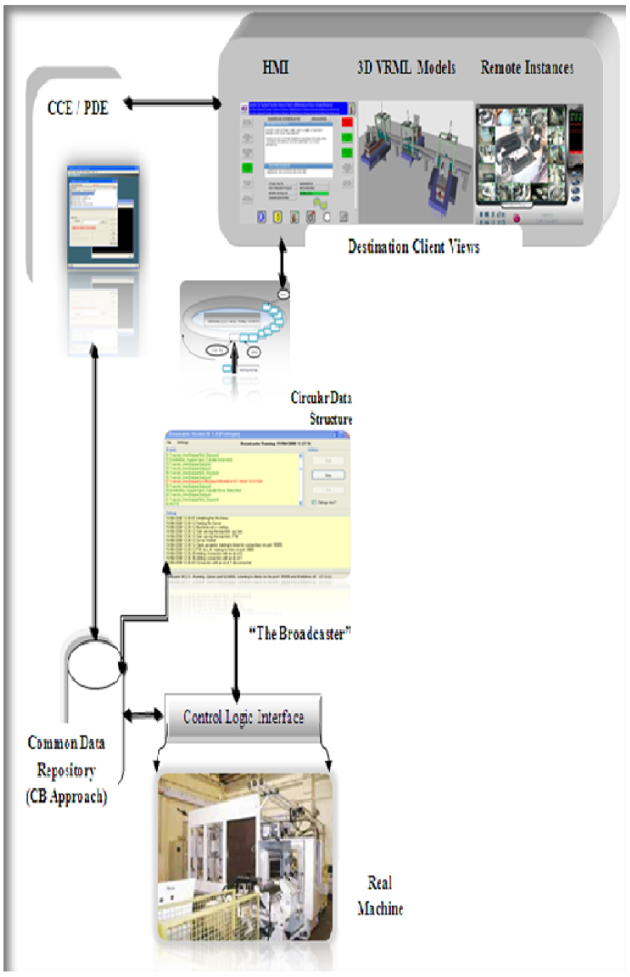

Fig. 2 CB engineering environments

III. DATA PROPAGATION MECHANISM

Distributing manufacturing activities does require an effective technique for capturing the shop-floor machine data in real-time, integrating and analysing it remotely using various enterprise resources having different mechanisms. In these circumstances, there is always a risk of collecting data faster than actually processing it and propagating it further or vice versa [8]. The prototype incorporates a circular-based data mechanism for propagating it to all the required manufacturing supply chain client resources. As data processing structures provide a method of storing data and propagating them as required, choosing the most efficient method reflects the provision of using the most efficient algorithm. A number such methods are available, some of which include non-linear processing structures (e.g. b-tree, decision tree) and linear processing structures (e.g. circular queue, linked list), just to mention a few [9]. With respect to the work carried out in this project, data needs to be stored in a volatile medium to allow fast constant-time data access for distributed supply chain clients resources. Storing data in a volatile medium is a method which is increasingly being used in many soft real-time applications. For propagating real-time machine information, authors have chosen a circular-based processing structure. This structure uses a bounded but reconfigurable size FIFO (*First In First Out*) buffer for storing and retrieving machine information. The items need not be shuffled when any position is being used in the buffer. With this approach, the semantics of the buffer routine is programmed in such a way that, when the buffer gets full, a subsequent writing operation starts on the oldest item in the occupied slot.

Other data processing mechanisms, for example, static linear queue suffers from a drawback such that when the queue is full, adding new information for storage cannot be done, even if one deletes some information. There was a strong need of having a solution in the prototype implementation, where the data could be saved such that when the "rear" reached the queue's limit, the "oldest" information automatically became the queue's new "rear" [9]. In soft real-time data propagation, time to allocate and reallocate memory in the system calls can cause a big hit to the performance metrics, especially when the incoming machine events are extremely fast. In the light of such a requirement, a circular-based data propagation structure offers more flexibility in terms of memory allocations, queue referencing, though they are tricky and complex to implement in a distributed environment. The biggest challenge identified during the initial implementation stage of this method was the problem of both the "Read" and "Write" function calls pointing at the same information when the queue was either entirely full or empty. In order to avoid this situation where both the pointers ("Read" and "Write" operation) were pointing to the same item (e.g. when the buffer was entirely full), authors implemented an additional global variable which kept a track of readable items in the queue. Any write process increments this variable and any read process decrements it accordingly. If both processes point to the same location, this variable distinguishes if the buffer is full or empty. The method adopted is simple; however, since the environment requires distributed simultaneous connections, multithreading approach has been implemented with a complex logic routine to keep track of the enqueue and dequeue processes. Authors have designed a collection of array-based logic routines and the hash mapping technique [9] to prepare messages that are to be processed by the mechanism and accessed asynchronously by remote client instances.

An algorithmic implementation for enqueuing and dequeuing is presented in the table I, where enqueuing corresponds to the addition of a new message into the rear of the queue, when the data propagates from the machine to the prototype, where as dequeuing corresponds to overwriting the oldest message from the front of the queue, when the queue reaches its preset limit. In the table I, "Mx" is the maximum preset capacity of the queue slots which can dynamically be reconfigured using a configuration tool available within the prototype itself. "S" corresponds to the various possible states in the algorithmic implementation and FV/RV corresponds to the position values of the front/rear pointers respectively. Fig 3 shows a diagrammatic representation of this implementation method. In the fig 3, a dedicated routine writes data into the next available slot in the circular buffer based on the algorithm described in the table I. Remote supply-chain client resources such as HMI, remote HMI, end user engineering tools, VRML models, etc, can establish a link to the prototype at any time to read the current state of the shop-floor machines. When all the immediate accessible slots have been occupied, the routine replaces the oldest item in the structure. This provides speedy soft real-time access for the connected clients.

TABLE I
DATA PROCESSING ALGORITHM

```
<Queue>=Full
S1:
If (FV_<queue> ≠ 0 AND RV_<queue> → Mx-1)   OR
(FV_<queue> → RV_<queue>+1) then
  ST: FULL

<Queue>=Empty
S2:
If (FV_<queue> = 0 AND RV_<queue>= FV_<queue> -1) then
   ST: EMPTY

<Queue>=Enqueue
S3:
 If !=S1 then
   If (RV_<queue> → Mx-1)then
      RV_<queue> = FV_<queue>
      S4
   Else
     RV_<queue> ++

   If (FV_<queue> → RV_<queue> AND
  Queue [FV_<queue>!=NULL]then
     ST: OVERFLOW
 Else
    Queue[RV_<queue>] ← Q_value

<Queue>=Dequeue
S4:
 If !S2 then
    If (FV_<queue> → RV_<queue>) then
    FV_<queue> =0 AND RV_<queue> = - 1)
    Else If (FV_<queue> → Mx-1) then
    FV_<queue> =0
    Else
    FV_<queue> ++
```

The dequeuing of machine information is carried out in a manner that allows multiple concurrent thread executions to access the queue for removing information without any conflict, and in a manner that does not excessively hinder operation of the implemented approach. The prototype is based on a heterogeneous design consuming a repository model and an event-driven invocation technique for efficient data propagation. Since the central repository, which represents the current state of the machine data, is the main trigger for selecting execution processes, the "Broadcaster" is based on the blackboard architectural design model. This architectural design offers the necessary data integrity and performance requirements needed in such a distributed manufacturing scenario. The "Broadcaster" has a dedicated broadcaster controller which handles real-time machine information publication / subscription via its interfaces in a circular buffer within the blackboard model. This controller syncronises and coordinates a smooth flow of access to the central repository of data (i.e. blackboard's circular buffer). A number of individual knowledge sources (internal as well as external) operate on the data structure. The external sources can be remotely implemented within the clients. The emplacement of the control is distributed i.e. implemented partly in the prototype's blackboard model for scheduling data access and partly in the external knowledge sources.

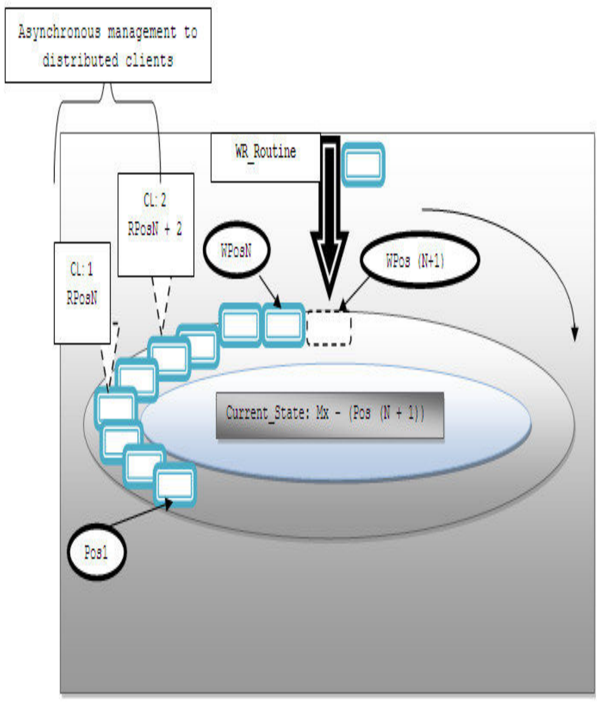
Asynchronous management to distributed clients
WR_Routine
CL: 2
RPosN + 2
CL: 1
RPosN
WPosN
WPos (N+1)
Current_State: Mx - (Pos (N + 1))
Pos1

Fig. 3 Circular-based data propagation

IV. EVALUATION

The reliability of any method is usually a measure of how well it operates in its environment. To gain confidence in the method adopted and to have a high quality outcome, one needs to resort to its evaluation. Furthermore, the performance of any method is an equally important issue which needs to be considered. With this in mind, the authors have carried out an evaluation study which involved development of software stubs set, acting as a group of source server and destination client(s). An overall test bed setup for evaluating the reliability and the performance of the adopted approach is shown in the fig 4.

The source server models the true system's behavior where the control device (e.g. An FTB) generates number of state-based, error-based and mode-based event machine messages and propagates them as packets over a TCP/IP based Ethernet network. "Broadcaster" is responsible to collect, format, process, sort, store and forward the data to various connected destination clients using publish-subscribe technique [10]. The maximum size of each transmitted data packet is currently set to 10 bytes which can include states, errors or modes of the machines. The structure and the format of the propagated data correspond to the structural representation within the Component-Based system architecture. Therefore the proposed solution assumes that the control system is engineered using the CB approach. The CB representation is the common system definition which stores all the machines configurations. It enables the visibility of the systems' information to all the globally distributed engineering partners [4]. The source server is scheduled to transmit data randomly ranging from tens of messages to hundreds of messages in a given time. By default, the size of the queue is 5000 items; however, it is reconfigurable via a local XML-based system configuration tool available within the prototype. The system is setup on a Windows-based platform, single CPU 2 GHz with 2 GB of physical memory.

The ultimate strategic aim of the research is to find flaws in the designed approach so that errors can be minimised / avoided, and ensure that the performance requirements are not compromised. From various past research projects held at Loughborough University, the average response times for data propagation to client resources such as a HMI is <500ms where as for remote clients, it is <1000ms for Powertrain assembly applications [11], [12]. This is the maximum propagation delay expected when a machine triggers an event, which has to be propagated to supply chain client resources. This analysis is beyond the scope of this paper; however, authors attempted to evaluate the reliability in terms of the operation by the system and its performance in terms of response times. The approach adopted by the authors for testing was to transfer all the possible data (valid as well as invalid machine information) using random class functionality offered by the dot NET environment. The evaluation was continuously run for approximately 1 hour daily for 5 consecutive days. The remote class generated all the possible machine information engineered by the CB representation of the demo machine (i.e. FTB-based web-service enabled Ford test rig). The incoming messages were individually time-stamped (using the *DateTime* structure) prior to their processing by the prototype and after their propagation. One of such test case (out of 50 test cases) is shown in the table II with ten average test results. The data structure size was kept between 500 and 5000 items only, where each item consumes up to 4 bytes of the assigned buffer slot. The time is shown in milliseconds, and t1 and t2 are implemented at the interface level of the prototype. It is clearly seen in the table II that the % loss of any packet is none, thus the reliability of the approach is unquestionable as none of the machine information was lost in translation via the adopted mechanism. In the table II and corresponding fig 5, the "mtx" (individual message processing time) value was approximately 74ms throughout.

This is the average processing time taken by the mechanism for propagating each client's event.

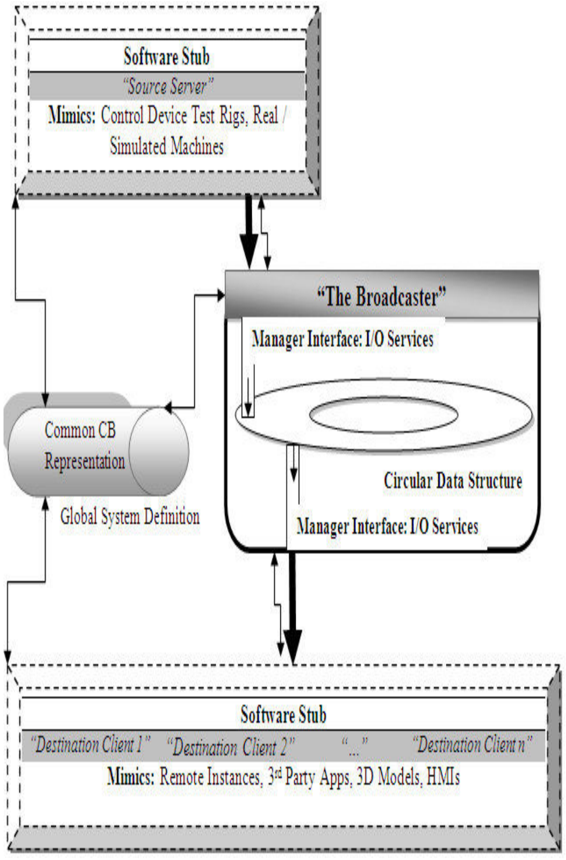
Software Stub
"Source Server"
Mimics: Control Device Test Rigs, Real / Simulated Machines
"The Broadcaster"
Manager Interface: I/O Services
Common CB Representation
Global System Definition
Circular Data Structure
Manager Interface: I/O Services
Software Stub
"Destination Client 1" "Destination Client 2" "..." "Destination Client n"
Mimics: Remote Instances, 3rd Party Apps, 3D Models, HMIs

Fig. 4 Evaluation test-bed set up

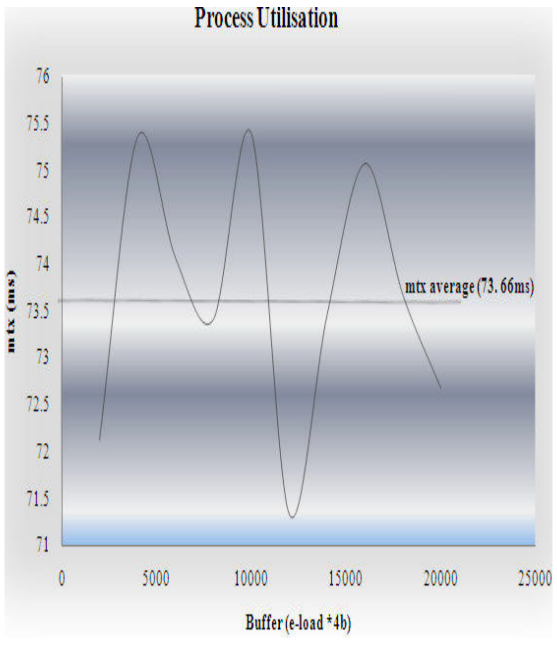

Fig. 5 Algorithm process utilisation

TABLE II
EVALUATION RESULTS

CID: T03	Data Processing Structure: **Circular**					
QId	E-Load	% Mem	t1 (ms)	t2 (ms)	tx (t2-t1) (ms)	Mtx (ms)
1	500	0.1	42250	78315	36065	72.13
2	1000	0.2	84500	159854	75354	75.35
3	1500	0.29	126750	237855	111105	74.07
4	2000	0.39	169000	315819	146819	73.41
5	2500	0.48	211250	399732	188482	75.39
6	3000	0.58	253500	467520	214020	71.34
7	3500	0.67	295750	552855	257105	73.46
8	4000	0.77	338000	638276	300276	75.07
9	4500	0.86	380250	711750	331500	73.67
10	5000	0.96	422500	785910	363410	72.68
➢ % Loss: **None**			Average **<u>73.66</u>**			
➢ Structure Status: **Full**			Buffer: E-load * 4b			

In order to justify the adopted solution, an alternative widely used data propagation mechanism (i.e. the linear queue) was compared with the current implementation for evaluation purposes only. For avoiding any effects caused by the network traffic, authors had used a single computer for this performance comparison. The linear queue's data propagation results showed a significant performance hit on the statistic metric side. When comparing both the approaches, circular-based data propagation implementation showed a much better performance in terms of response times than a linear one. This can easily be seen from the fig 6, where there is a difference between computational response times when using two different approaches of manufacturing data propagation. This research has found out that the performance in terms of response time offered by implementing the circular-based mechanism for propagating manufacturing information is much better than the linear mechanism by up to 87% approximately.

The response time of the linear queue to propagate a machine event was found to be approximately 550 ms, which is more than the circular implementation, when the buffer was full. This difference is due to the buffer management costs involved when the entire buffer has been filled up with data. This results the linear queue to perform relocation before accepting any new request to join the waiting queue. Thus, linear queue is affected by the queue length when making relocation (i.e. restoring the location of the queue). On the other hand, circular approach just needs to modulo the queue length and point to the buffer address to handle the requests, thus less management costs are involved. Overall, linear queue performs more operations than the circular approach, resulting into the linear queue to spend more time handling propagation requests comparatively.

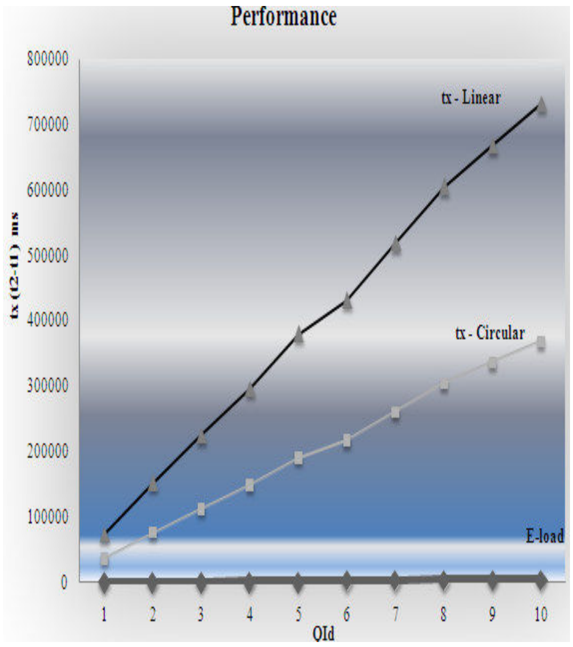

Fig. 6 Performance Comparison

V. CONCLUSION

This paper has focused on the implementation of a an efficient real-time data propagation mechanism for distributing manufacturing machine information to a number of supply chain remote clients such as a HMI, VRML-based 3D models and other client instances, regardless of their geographical locations or system mechanisms. The circular-based mechanism has been implemented in the prototype "Broadcaster" which has been developed at Loughborough University with a clear aim to solve the problem of distributed data propagation faced in manufacturing environments. The main concentration of the work has been to evaluate the reliability and the performance of the implemented approach. Results have shown that the reliability of this approach is unquestionable as there is no data loss when the machine information is transmitted from its source to its destination via this prototype.

An alternative data propagation mechanism (i.e. linear queue) has been evaluated to compare the results with the circular-based approach. Evaluation results have shown that the performance of circular approach is far better (87% more) than the linear queue in manufacturing data propagation. With the circular approach, free buffer space is always available to have up-to-date data appended to it in real-time, thus the pointer is adjusted accordingly. There is no need of expensive processes like relocation or copying of the data across the buffer space, as these tend to increase the management costs resulting into consuming more processing time.

These are obviously a set of laboratory results obtained as further research work is to be carried out when implementing such a solution to a real manufacturing environment like a Powertrain assembly line. The immediate research scope for further work surrounds the scalability factor of the adopted approach, for example, the throughput in terms of the number of machines it can handle, potential buffer overflow scenarios involved when connected remote clients are very slow, having intermittent connections etc. The results of these researches are going to be published.

ACKNOWLEDGEMENTS

Authors would like to gratefully acknowledge the support obtained for this work from the EPSRC, Loughborough IMCRC and EU SOCRADES framework 6 research projects. The research contributions made by other members of School of Mechanical and Manufacturing Engineering and industrial collaborators are also fully acknowledged.

REFERENCES

[1]. McFarlane, D.C. and S. Bussmann, *Holonic manufacturing control: rationales, Developments and Open Issues.* Agent Based Manufacturing: Advances in the Holonic Approach, 2003.
[2]. Mellor, E.W., R. Harrison, and A.A. West, *Reconfigurable user interface's to support monitoring and diagnostic capabilities within agile automated manufacturing system's.* Robotics, Automation and Mechatronics, 2004 IEEE Conference on, 2004. **1**.
[3]. Lee, S.M., R. Harrison, and A.A. West, *A component-based control system for agile manufacturing.* Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture, 2005. **219**(1): p. 123-135.
[4]. Harrison, R., A.W.Colombo, A.A.West and S.M.Lee, *Reconfigurable modular automation systems for automotive power-train manufacture.* International Journal of Flexible Manufacturing Systems, 2006. **18**(3): p. 175-190.
[5]. Harrison, R., A.A.West, R.H.Weston and R.P.Monfared, *Distributed engineering of manufacturing machines.* Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture, 2001. **215**(2): p. 217-231.
[6]. Ong, M.H., A.A.West, R.P.Monfared and R.Harrison, *Application of enterprise modelling technique for specifying a component-based system.* Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture, 2005. **219**(9): p. 649-664.
[7]. Harrison, R., S.M. Lee, and A.A. West, *Lifecycle engineering of modular automated machines.* 2nd IEEE International Conference on Industrial Informatics, 2004: p. 501-506.
[8]. Sommerville, I., *Software engineering*. 8th ed. 2007, Reading, Massachusetts: Addison-Wesley Publishing Company.
[9]. Tarjan, R.E., *Data structures and network algorithms*. 1983: Society for Industrial Mathematics.
[10]. Linthicum, D.S., *Enterprise application integration*. 2000.
[11]. Lee, L.J., *A next generation manufacturing control system.* PhD dissertation, MSI Research Institute, Loughborough University (internal), 2003.
[12]. COMPAG, *COMponent Based Paradigm for AGile Automation.* Loughborough University research project.