1. Introduction

The increasing use of distributed client/server computing arrangements among geographically dispersed teams demand new network architectures capable of providing higher bandwidths (e.g., 45 Mbps and above) and a LAN-like performance over longer distances.

In this paper, we analyze the performance of two new high-speed multiple-access network standards, the Fiber Distributed Data Interface (FDDI) and the Distributed Queue Dual Bus (DQDB). The performance issues for FDDI and DQDB have received considerable attention lately [1],[2] and [3]. The papers have primarily focussed on performance issues related to the protocol behaviour without adequately accounting for all the requirements and traffic characteristics of end-user applications. For instance, the unfairness aspects of the DQDB protocol have been considerably highlighted in [2],[3]. However, most of the analysis related to unfairness of DQDB has been implicitly carried out only in the context of access delays and bandwidth allocation in bulk data transfer environments. In many other practical situations, e.g., mixed interactive and file transfer applications, and distributed client/server applications, DQDB turns out to be a highly appropriate and fair solution as we demonstrate in this paper.

Although FDDI and DQDB networks are both capable of supporting synchronous (e.g., voice) and asynchronous (e.g., bursty data) applications, we focus only on asynchronous applications. The paper briefly describes the two protocols, compares their bandwidth efficiency, delay performance, and fairness with respect to delay and bandwidth allocation, and shows what applications can benefit from the characteristics of each protocol.

2. The DQDB Protocol

The DQDB media-access protocol proposed in the IEEE 802.6 standard uses an architecture based on a pair of 150 Mbps contra-flowing unidirectional buses. Both buses operate with fixed length slots allocated to either synchronous or asynchronous traffic. The head station generates frames containing several slots. Slots for synchronous traffic are reserved through agreement with the head station. The access to the bus by asynchronous traffic is governed by a distributed queueing algorithm that attempts to order the access to the bus according to the time of arrival of messages at different stations. The access mechanism is such that when a station receives a message to transmit, the source and destination MAC addresses are appended to it forming a packet. This packet is broken into several fixed-length segments called "cells," and protocol header fields are appended to each segment, thereby forming segments the size of a slot.

Because of the small segment size, short messages have quick access to the bus, while long messages uniformly share the bus bandwidth. The price of having small segment sizes is that the total amount of overhead in the system is relatively high. It is a round-robin like service policy; the quantum of service at every station is of, at most, a segment of data, and all active stations tend to have a quantum of service before any particular station is served twice. Each station has the responsibility of allocating the available bandwidth (the empty slots that are passing by) between itself and stations downstream of it. The mechanism can be described as follows. A particular station positions itself in an idealized single-server queue, based on the information of how many stations downstream have issued a request. Every station is allowed to have at most one outstanding request. Every time a particular station is served and still has something to transmit, it is allowed to issue another request. Thus, stations that have been served earlier than a particular station are likely to have issued another request before that particular station is served. A comprehensive description of the protocol operation can be found in [4],[5].
is reset to TTRT and enabled again. The FDDI allows for two types of traffic - synchronous and asynchronous. For synchronous transmission, each node is pre-assigned an amount of time called synchronous bandwidth allocation, each time it receives a token. The total of all synchronous assignments is not to exceed TRT. When a token arrives at a station the TRT is set to TTRT and begins to count down. When the token arrives during the next rotation and if TRT has not expired, then the current value of TRT indicates the difference between the actual token rotation time and TTRT. This value is loaded into another timer called Token Holding Timer (THT) which then starts to count down. The station first transmits its synchronous traffic for its assigned duration. However, the station is allowed to transmit asynchronous frames only for the difference between Actual Token Rotation Time (ATRT) and TTRT. The station sends asynchronous traffic until THT expires.

If no limits are placed on the amount of time a station can transmit asynchronous frames (exhaustive service), then a heavily loaded station will tend to monopolize the bandwidth. In order to avoid such occurrences, the token protocol has been introduced and on any given rotation a station can only monopolize an amount of bandwidth equivalent to the slack between ATRT and TTRT. The smaller the TTRT, the smaller the opportunity for a station to hog the network’s bandwidth. However, decreasing TTRT necessarily reduces average token rotation time. Depending upon network loading conditions, more than one rotation may be needed for serving all stations and during each one of these rotations the propagation delay incurred by the token is essentially wasteful and leads to inefficient utilization of network bandwidth. We therefore observe that there are two opposing forces at work here. Decreasing TTRT tends to make the access protocol “fairer,” however, it makes the network more inefficient. A comprehensive description of the protocol operation can be found in [6].

4. Performance Issues

The performance measure that we consider here is "fairness" in the context of message delays as a function of the message size distribution and the relative load offered by different stations. Fairness is not a well defined concept; one has to be specific in describing the particular aspect that is being considered. Identical stations in FDDI are treated identically since it is a symmetrical system. Based on this concept we can say that FDDI is fair. However, when the system operates under unbalanced load, strange things can happen. For example, stations with higher loads could experience smaller delays than stations with lighter loads. In some sense, this is a form of unfairness. Sometimes a short packet may have to wait for a preceding station to send all its packets before it can be transmitted on the ring. This results in longer delays for short packets. This could be viewed as another aspect of unfairness.

In the case of DQDB, the scheduling of different stations gets distorted as the distance and number of stations increase, generating discrepancies between access delays even for identical stations. This results in an unfair bandwidth allocation under heavy traffic conditions [2],[3]. However, since DQDB uses both buses for data transmission, discrepancies in access delays may be compensated by other delays in the data path. The appropriate measure for delay is extremely dependent on the applications and their traffic characteristics. For example, stations that are further downstream on a particular direction of transmission are the ones that suffer the most from scheduling discrepancies, and at the same time, they are in average closer to destination stations. Therefore, from a total end-to-end delay point of view all stations see nearly the same performance and there is no unfairness in this case.

We present a comparative delay analysis of the two systems by estimating delays for short and long messages under balanced and unbalanced traffic conditions. This comparison is relevant from a practical point of view since most data applications show a traffic distribution that tends to be bimodal (i.e., frequency of message sizes have peaks at low and large values).

5. Analytical Models

5.1 DQDB Model

Consider a DQDB Bus of length \( L \) miles interconnecting \( N \) stations (each station adds \( \tau_1 \) of latency) which are uniformly distributed throughout the length of the Bus, as depicted in Figure 1. Let \( h \) bytes of header and \( d \) bytes of payload constitute a cell, which fits in a slot. Let \( 1/\mu \) denote the Bus speed in slots per second in each direction of transmission. The arrival process of messages to the \( i^{th} \) station, where \( i \in \{1, ..., N\} \), is Poisson with arrival rate \( \lambda_i \) and message distribution \( f_i(k) \) with mean \( \bar{m} \). Let the probability of station \( i \) transmit a message to station \( j \) be given by

\[
P_{i,j} = \begin{cases} 0 & ; j = i \\ \frac{(1-p)^{|i-j|} - p}{1 - (1-p)^{N-i}} & ; j \neq i \\ \end{cases}
\] (1)

where the value of parameter \( p \) corresponds to a measure of traffic locality.

One can calculate the efficiency of a DQDB network due to segmentation into cells (since some cells are partly filled), by dividing the average carried user data by the corresponding average Bus traffic.

\[
Efficiency = \frac{\sum_{i=1}^{N} \lambda_i \sum_{k} \frac{k}{d} (d + h) f_i(k)}{\sum_{i=1}^{N} \lambda_i \sum_{k} \frac{k}{d} f_i(k)}
\] (2)

Here, \([x]\) represents the integer part of the real number \( x \).

The DQDB protocol is a work conserving discipline with respect to the Bus traffic (i.e., segmented messages). When distances are small, the MAC discipline in each direction of a DQDB Bus behaves like a processor-sharing server where active stations receive a quantum of service corresponding to one slot in each "cycle". (A "cycle" corresponds to serving all active stations). As the Bus length gets large, the MAC discipline becomes a "distorted" processor-sharing discipline (i.e., stations upstream receive more service per cycle than stations downstream. Thus, we can still estimate the network average message delay for all Bus lengths based on conservation laws of generalized processor-sharing systems [7]. One interesting characteristic of DQDB (and of processor-sharing disciplines in general) is that it is very robust to changes in the burstiness of background traffic; the mean delay performance is essentially determined solely by the average Bus utilization. The expression for the network average message delay is similar to that of an
M/M/1 queue and is given by,

\[ E[D_{\text{un}}] = \frac{m_{\rho}}{\mu(1-\rho)}, \]  

where \( \rho \) denotes the unidirectional Bus utilization. Although the processor-sharing model becomes exact in the limit as the number of nodes and message sizes become large, it can be shown that this model corresponds to a worst case analysis of the actual system's total average message delay. The expected message delay as a function of message size can also be estimated based on the generalized processor-sharing model. The expected delay on the network for a message of length \( M \) is given by,

\[ E[D(M)] = \frac{M \rho}{\mu(1-\rho)}. \]  

This particular result for generalized processor-sharing becomes inaccurate when the number of stations is very small or if a small number of stations generate most of the load.

To characterize delay as a function of position on the Bus, we consider two extreme cases. The first one assumes the most unfair distribution of bandwidth possible due to station latencies and propagation delays, which corresponds to stations upstream having preemptive-priority over stations downstream. This model is a worst case scenario for the effects of distortions no matter how large the total latency and propagation delays are. The second one assumes another distortion on the perfect scheduling, which corresponds to stations with lighter load receiving proportionally more service than stations with higher load. This model is a worst-case scenario when the total latency and propagation delays are small.

We derive an approximation for the average message delay as a function of position on the Bus for the unfair behavior. We model the system, for each direction on the Bus, as a preemptive head-of-the-line priority system where a station is assigned a priority higher than those stations downstream from it. The preemptive-priority nature of the model tends to accentuate the unfairness due to distortions on DQDB's scheduling discipline. The expected message delay for station \( i \) on a given Bus is given by the respective preemptive-priority discipline delay equations.

\[ E[D_i] = \frac{\overline{m}_i}{\mu} + \frac{\sum_{k=1}^{i} \lambda_k X_{k,i}^2}{2 (1 - \gamma_{i-1}) (1 - \gamma_i)}, \]  

where \( \gamma_i = \sum_{k=1}^{i} \rho_k \) denotes the load generated by station \( i \) and all stations upstream from station \( i \), \( \rho_k = \sum_{j=k+1}^{N} \lambda_j m_j P_{k,j} \) (or \( \rho_k = \sum_{j=k+1}^{N} \lambda_j m_j P_{k,j} \) for upstream direction) denotes the load generated by station \( k \) in a given direction and \( X_{k,i}^2 \) denotes the second moment of the service time of a message at station \( k \).

Next we derive an approximation for the average message delay as a function of position on the Bus for the perfect scheduling case. The system serves the stations uniformly, however, the load on a certain Bus direction from each station is a function of the station's position on the Bus. Therefore, the actual system approaches a non-symmetric head-of-the-line processor sharing system which is difficult to analyze. Thus, we analyze a multi-class preemptive priority system where a station participates in several classes depending on its load and the load of others in the system. This auxiliary system differs from the actual system in the sense that it further benefits the stations with smaller loads. Based on the load generated by each station in one direction of transmission (i.e., \( \rho_k \)), we rank the stations from 1 (i.e., the station with the smallest load) to \( N \) (i.e., the station with the highest load). All stations in the auxiliary system participate in the highest priority class with a load identical to the lowest ranked station (i.e., \( \rho_{[N]} \)). All stations but the lowest ranked one participate in the next to highest priority class with a load identical to the increment from the lowest ranked station to the next ranked station (i.e., \( \rho_{[N]} - \rho_{[1]} \)). In a similar way, one can find the participation of all stations in all other priority classes. The preemptive-priority nature of the auxiliary system tends to accentuate the delay difference due to DQDB's intrinsic load unbalance on stations on each Bus. The expected message delay for station \( r[i] \) on a given Bus is given by the following equation.

\[ E[D_{\text{un}}(i)] = \sum_{k=1}^{i} T(r[k]) \frac{\overline{m}_i}{\mu} + \frac{\sum_{j=k+1}^{N} \rho_{[j]} \sum_{n=1}^{j} X_{n,j}^2}{2} \]  

\[ T(r[k]) = \frac{\sum_{j=k+1}^{N} \rho_{[j]} \sum_{n=1}^{j} X_{n,j}^2}{\mu (1 - \sigma_{i-1})} + \frac{\sigma_i (N-j+1) \alpha(r[j])}{(1 - \sigma_{i-1}) (1 - \sigma_i)} \]  

where \( \sigma_{i} \) denotes the number of the \( i \)th ranked station, \( \sigma_{i} = N - j + 1 \) (or \( \sigma_{i} = N - j + 1 \) for upstream direction) denotes the traffic generated with priority \( i \) and higher, \( \alpha(r[j]) \) denotes the incremental load generated by station \( r[j] \) with respect to station \( r[j-1] \), and \( X_{n,j}^2 \) denotes the second moment of the service time of a message at station \( r[k] \).

The actual distortions on DQDB's scheduling mechanism will always be milder than what is predicted by the above models.

5.2 FDDI Model

We will first briefly describe the modeling approach and subsequently describe the details. At each station, messages are assumed to arrive according to a Poisson process. The packet size distribution is assumed to be general. Packets are served only when the station obtains the token and if there is slack between ATRT and TTRT. Packets are served for a duration equal to the slack between ATRT and TTRT. Upon expiry of this slack the token leaves to serve other stations. This is equivalent to the server taking a vacation from the point of view of a single station. Depending upon the network loading conditions, on the next rotation, there may be no slack between ATRT and TTRT. Until there exists such a slack the queued packets do not get service. If slack exists when the token arrives at a station then the vacation ends. Now the station will get service for a limited period equal to the slack available. Therefore the overall methodology that we apply here is an M/G/1 vacation time model with the additional constraint of limited service. Such models have been studied widely in the queueing literature \[16, 20, 26\]. We will apply those results here. The key problem is to identify the vacation
time parameters as well as to account for the service time constraints. We focus on our attention on these two aspects.

First, we study the case of two stations on the FDDI ring as shown in Figure 2-b. Let \( c_1 \) be the propagation time from station 1 to station 2 and \( c_2 \) be the propagation time from station 2 to station 1. We assume that \( c_1 \) also includes the latency at station \( i \). Let \( c_1 + c_2 = W \), the total token rotation time under no load. The timing diagram for this configuration under saturated traffic conditions is shown in Figure 2-a. We have assumed that packets are queued up and waiting to be served whenever the token arrives to give service to the station. In the figure \( a_i \) represents the arrival instant of the token at station \( i \) and \( d_i \) represents the departure instant of the token at station \( i \).

In Figure 2-a, we assume that in the first rotation there is no load on the ring and from the beginning of the second rotation a saturation load is applied to each of two stations. From the figure it can be seen that between token arrival instants at station 1 during second and third rotations, a time equal to TTRT has elapsed. Therefore on the third rotation of the token, packets queued at station 1 will not be served. On the third rotation, packets queued at station 2 will be served for a duration equal to TTRT - W. On the fourth rotation neither station 1 nor station 2 will be served since the actual token rotation times in preceding rotation are equal to TTRT at both stations. Therefore rotation four is identical to rotation one and the cycle of events repeats. The cycle consists of three rotations and the cycle duration is \( 2 \cdot \text{TTRT} + W \). Let the mean arrival rate at each station be \( \lambda/2 \) packets per second and the mean packet service time be \( \overline{S} \). Therefore, the mean number of packets arriving at each station in a cycle is, \( \lambda(2 \cdot \text{TTRT} + W)/2 \), and the mean number of packets served in each cycle is, \( \text{TTRT} - W/\overline{S} \).

For stability, the mean number of packets arriving during a cycle must be less than the mean number of packets that can be served. Therefore, the maximum utilization, \( R_{\text{max}} \), that can be supported on FDDI before queues build up indefinitely is given by,

\[
R_{\text{max}} \leq \frac{2(\text{TTRT} - W)}{2 \cdot \text{TTRT} + W}.
\]

Using the same general logic, it can be shown that for \( N \) symmetrically placed stations on the ring, \( R_{\text{max}} \) is given by,

\[
R_{\text{max}} \leq \frac{N(\text{TTRT} - W)}{N \cdot \text{TTRT} + W}. \tag{7}
\]

Let \( c \) be the mean token rotation time. Since \( c \) is independent of the access protocol, it can be shown that,

\[
c = \frac{W}{1 - \rho}, \tag{8}
\]

where \( \rho \) is the load on the ring. The mean token rotation time attains its maximum value when a load of \( R_{\text{max}} \) is applied. Therefore \( c_{\text{max}} \) is obtained by substituting \( R_{\text{max}} \) in the above equation,

\[
c_{\text{max}} = \left( \frac{N}{N + 1} \right) \text{TTRT} + \frac{W}{N + 1}. \tag{9}
\]

As described before, we have seen that a station does not necessarily receive service on every token rotation and that the cycle duration could be as long as \((N + 1)\) token rotations under saturation conditions. When the ring is lightly loaded the cycle duration can be one rotation. We hypothesize that the cycle duration can vary between 1 and \((N + 1)\) rotations depending upon the offered load. We assume that when the cycle duration is \( (k + 1) \) rotations, the cycle time is \( k \cdot \text{TTRT} + W \). Therefore, we have

\[
(k + 1)c = \frac{(k + 1)W}{1 - \rho} = k \cdot \text{TTRT} + W.
\]

Solving the above for \( k \),

\[
k = \frac{\rho W}{(1 - \rho) \text{TTRT} - W}. \tag{10}
\]

Let \( \lambda_i \) and \( \overline{X}_i \) be the mean packet arrival rate and mean packet service time respectively at station \( i \). In equilibrium, the service given to each station during a cycle is equal to the load offered at the station. Let \( T_i \) be the mean data transfer time at station \( i \) during the cycle. We have,

\[
\overline{T}_i = \rho_i(1 + c_1)\overline{V}_i, \tag{11}
\]

where \( \rho_i = \lambda_i/\overline{S}_i \). The mean vacation time is the difference between the mean cycle time and the mean data transfer time. Denoting \( \overline{V}_i \) to be the mean vacation time at station \( i \), we have

\[
\overline{V}_i = (k + 1)c(1 - \rho_i). \tag{12}
\]

Next we use an upper bound for mean delay in an M/G/1 vacation time model with limited service derived in [8]. The upper bound for mean waiting time at station \( i \) is given by,

\[
E[D_i] \leq \frac{\lambda_i E(S_i) + (1 - \rho_i) \frac{E(V_i)}{\overline{V}_i} + \frac{2\rho_i \overline{V}_i}{\overline{S}_i} - (x_i - 1)\lambda_i \overline{V}_i / x_i}{2(1 - \rho_i - \lambda_i \overline{V}_i / x_i)}. \tag{13}
\]

where \( S_i \) is the random variable representing packet service time at station \( i \), \( V_i \) is the random variable representing vacation time at station \( i \), and \( x_i \) is the maximum number of packets served per visit at station \( i \). \( x_i \) is determined by TTRT and is given by,

\[
x_i = \frac{\text{TTRT} - W}{\overline{S}_i}. \tag{14}
\]

The delay upper bound given above essentially depends on the first two moments of vacation time. We now discuss a method to approximate the variance of vacation time. We observe from Figure 2-a that under saturated loading conditions, the cycle time is constant and hence the variance of vacation time is zero when \( \rho = R_{\text{max}} \). The minimum value \( V \) can attain is \( W \) and this can happen when there is no load on the Ring which can occur due to the probabilistic nature of packet arrivals. We will now assume that \( V \) is
uniformly distributed between $W$ and $Z$ such that $E(V) = \frac{V}{2}$, given above. Therefore, $Z = 2V - W$.

The assumption that $V$ can attain a minimum value of $W$ is certainly conservative. The uniform distribution assumption implies that the minimum value is attained with the same probability as the mean and is therefore also conservative.

Next, from the assumption of uniform distribution of $V$ and the fact that vacation time is constant at $p_{\text{max}}$, we approximate the variance of vacation time $\sigma_v^2$ by:

$$
\sigma_v^2 = (p_{\text{max}} - \beta)^2 \left( \frac{Z^2 - W^2}{12} \right) \quad (15)
$$

where $\beta$ is a constant. In performing delay calculations, we have varied $\beta$ between 0.5 and 2.0, and the variation in delay estimate has been less than 4%.

6. Results and Discussion

Performance sensitivity of DQDB and FDDI to distance, number of stations and traffic characteristics are distinctively different. For example, as distance and number of stations increase, the bandwidth efficiency of DQDB does not get affected but fairness (in terms of equal access to all stations) degrades. The exact opposite effects occur with FDDI; as distance and number of stations increase, fairness remains unaffected while bandwidth efficiency degrades. DQDB efficiency depends only on the MAC header overhead, the cell overhead (header and adaptation layer fields), and the packet segmentation/reassembly overhead. FDDI efficiency depends on the MAC header overhead and on the token rotation time. Since distance and number of users affect token rotation time, they also affect FDDI efficiency.

For example, if $N$ stations are active and the no load token rotation time is denoted by $W$, the normalized station throughput can be approximated by $(TTTR - W)/(N \cdot TTTR + W)$. As distance increases, $W$ increases, resulting in a decrease in throughput. Similarly, as the number of stations increases, $W$ increases due to increased station latency, resulting in a decrease in throughput. Since the value of $TTTR$ can be tuned, a compromise (e.g., between delay and throughput) can be reached based on some acceptable performance criteria.

Based on the analytical models developed here, we have computed delay performance measures for both FDDI and DQDB. The comparison highlights how the two protocols treat short and long messages in a mixed applications environment as well as the unfairness caused by the presence of heavy traffic stations in an unbalanced load environment characteristic of distributed client/server arrangements.

In all computations we assume that the FDDI Ring speed and the DQDB Bus speed are both 100 Mbps. The number of stations is held fixed at 100. The stations are spaced uniformly apart on the Ring/Bus. The DQDB cell header size is assumed to be 5 bytes, the adaptation layer overhead is assumed to be 4 bytes per cell, and the cell payload size is assumed to be 44 bytes in accordance with the IEEE 802.6 specifications [4].

In FDDI, we know from equation (7) that, for large values of $N$, $p_{\text{max}} = 1 - (W/TTTR)$. Therefore $TTTR$ needs to be chosen so that it is sufficiently large relative to $W$ so that the ring does become unstable at low loads. However, if $TTTR$ is chosen to be very large then the cycle time begins to lengthen at higher utilizations and individual stations get service after longer vacation times. Bearing these factors in mind, we have chosen $TTTR$ to be 10 msecs for a Ring of length 100 miles, and 5 msecs for a Ring of length 1 mile. When $TTTR$ is 10 msecs for a Ring of 100 miles and 100 stations, the maximum load $p_{\text{max}}$ is 92%. In this case, the maximum cycle time is 1.008 seconds and every station will receive service within the cycle time. For a Ring of 1 mile and 100 stations, a $TTTR$ of 5 msecs keeps $p_{\text{max}}$ at 99.8% and the maximum cycle time at 0.5 seconds.

Figures 3 and 4 show the mean packet (i.e., message) delay in a mixed application environment consisting of both interactive and file transfer traffic. Two different Ring/Bus lengths are considered: 1 mile (Figure 3) and 100 miles (Figure 4). All stations are assumed to be uniformly loaded. The traffic mix at each station is assumed to be bimodal consisting of 1500 byte "long" packets representing file transfer traffic and 64 byte "short" packets representing interactive traffic. 20% of all packets are assumed to be long and the remaining 80% are assumed to be short. The mean packet size turns out to be 350 bytes. The figures show that irrespective of the distance involved, DQDB gives a significant delay advantage to short packets relative to the long ones. On the other hand, short packets see smaller delays than long packets on an FDDI network only when the distances are small and the network is lightly loaded ($\rho < 0.5$). At high loads and/or long distances, FDDI networks are unable to give preferential treatment to short messages. Therefore, DQDB is a better alternative in an environment dominated by interactive applications. For file transfer traffic involving long messages, FDDI provides lower delays than DQDB when the distances involved are small; however, their performance becomes comparable as distances increase. The DQDB networks saturate faster due to the increased overhead involved.

Figures 5 and 6 show the impact of a heavy user on the delays experienced by light users in an unbalanced client/server type of an arrangement. The 50th station is assumed to be the heavy user, i.e., a server, while the rest are assumed to be light users, i.e., clients. The packet sizes at all stations are exponentially distributed with a mean of 350 bytes. As the load contributed by the heavy user is varied from 20% to 90%, the load contributed by the light users is appropriately adjusted to keep the overall Ring/Bus utilization fixed at 70%. Again, two different Ring/Bus lengths are considered: 1 mile (Figure 5) and 100 miles (Figure 6).

At short distances, the DQDB network behaves as a perfect scheduling system and therefore the packet delays for light users are significantly smaller than those for the heavy user. Furthermore, as the heavy user contributes more and more to the network traffic, the delays experienced by light users continues to decrease. As most of the load concentrates at the heavy user, the network behaves increasingly as FIFO for the heavy user, and therefore, the heavy user's packet delays also decrease. It should be remembered that the network average delay remains constant irrespective of the load shift across the stations. At longer distances, the load contributions caused by significant propagation delays dominate the impact of unbalanced traffic. In such situations, the heavy user has very little impact on either its own delays or the delays seen by the light users.

In the case of FDDI, however, we see that the result is quite the opposite. The heavy user clearly has an advantage because its delays are smaller than those of the light users. Furthermore, as the heavy user contributes more and more traffic to the system, it reduces its packet delays at the
expense of higher delays experienced by the light users. These effects are accentuated as the distance increases. Therefore, in the sense of protecting light users from the heavy ones, DQDB is fairer than FDDI. The results of Figures 5 and 6 demonstrate that, from an applications point of view, DQDB is a preferred alternative, when compared to FDDI, for distributed client/server computing applications.

Figure 7 shows the message delay at a station as a function of the station’s position on the DQDB Bus. The message size is assumed to be exponentially distributed with mean 350 bytes at all stations. All stations are assumed to offer the same load to the network. As the distance increases, the delays seen by stations on the middle of the Bus increases while the delays seen by stations towards the edges of the network decreases.

Proper network engineering will require the appropriate choice of Bus length depending upon the number of stations and traffic distribution so as to provide uniform access delays to all stations. This type of unfair access delay behavior has been observed in other studies as well [2]. For obvious reasons, the message delay at a station on a FDDI network is independent of the station’s position on the Ring.

7. Conclusions
We have presented a performance comparison of the DQDB and FDDI multiple-access protocols that are designed for high speed operation over local and metropolitan area distances. The analysis performed here accounts for the intrinsic characteristics of data traffic encountered in mixed applications environments and in distributed client/server computing environments. We conclude that DQDB performance is superior to FDDI under most realistic MAN environments. FDDI proves to be superior only when distances and number of stations are small (e.g., in a computer room) and when the applications mostly require the transmission of long messages while DQDB proves to be superior for distributed computing environments and in mixed traffic environments when the delay requirements for interactive messages are very stringent.

REFERENCES
Comparison of Short and Long Packet Delays in FDDI and DQDB
Length = 1 mile

Comparison of Short and Long Packet Delays in FDDI and DQDB
Length = 100 mile

Comparison of Delays for Light and Heavy Users in FDDI and DQDB
Length = 1 mile

Comparison of Delays for Light and Heavy Users in FDDI and DQDB
Length = 100 mile