DECelms - Managing Digital's FDDI and Ethernet Extended Local Area Networks

By Bruce E. Sweet

Abstract

The DECelms software product provides extended local area network management for Digital's Ethernet/IEEE 802.3 and fiber distributed data interface (FDDI) bridges and for its FDDI wiring concentrator. Product development entailed keeping pace with a changing set of requirements. These included the evolving ANSI FDDI standard, the proposed Digital Network Architecture FDDI data link specification, the Enterprise Management Architecture, the ability to extend the serviceability of the products, and the aggressive schedules of the hardware and firmware development teams. DECelms development resulted in an improved network management functionality including fault, performance, and topology management. These advanced features required corresponding enhancements as a part of Digital's FDDI program.

DECelms Development

The DECelms product, Digital's extended local area network (LAN) management software, provides remote network management for Digital's LAN Bridge 100, LAN Bridge 150, LAN Bridge 200, DECbridge 500, and DECconcentrator 500 products. The remote networks are included in the extended LAN by means of transparent bridges. DECelms functionality includes basic SET and SHOW capabilities, fault management, performance monitoring, FDDI ring mapping, automatic device discovery, and user alarms. The DECelms software runs as an application on a VAX processor running under the VMS operating system.

When Digital embraced the new fiber distributed data interface (FDDI) LAN technology, the role to the user interface and dependable documentation. The development team met these challenges and successfully delivered the DECelms product to market

that network management would play in the first product set was unclear. As our understanding of the technology grew, and we recognized the differences between the token ring architecture and the carrier sense multiple access with collision detection (CSMA/CD) protocol, the challenge became obvious. Both technical and business reasons made network management a priority.

On Ethernet/802.3 LANs, network management tools provide the ability to monitor the networks and troubleshoot problems as they arise. The failure mechanisms of Ethernet /802.3 LANs, such as faulty transceiver taps, continuously transmitting stations, and broadcast storms, are well understood. In addition to network management tools, powerful datascopes aid the network manager in recognizing and correcting these problems. Moving into the 100-megabit (Mb) world of the FDDI token ring brought a new set of problems, some of which were understood and others not even imagined. We realized the need to offer a network management solution that was capable of performing the same functions as our Ethernet/802.3 products and had the additional functionality necessary

to meet the new challenges of the FDDI technology. not only a good network management solution but a key development tool as well.

Key FDDI Differences

Several key differences between the FDDI and Ethernet technologies determined the specific requirements of the DECelms product. First, the physical topology and the location of stations attached to an FDDI ring play a significant role in how the ring operates. Each station is active and must participate in connection management to form a working LAN.[1] On Ethernet/802.3 LANs, each station is passive until it wishes to use the network. Thus, an Ethernet/802.3 LAN can operate with stations that do not strictly observe the protocol, since less stringent protocol requirements between stations are necessary to make the LAN work. This difference places a high priority on the ability to manage topology, a functionality that is less significant on Ethernet /802.3 LANs. Requirements to build FDDI ring maps and examine third-party station management (SMT) frame data grew from this priority.

A second difference is the need to manage the FDDI

Further, no FDDI datascopes were available to aid in the development of the FDDI product set. To correct this deficiency, requirements for DECelms functionality called for the software to be physical layer. This need arose mainly because the DECconcentrator 500 product is a physical layer device and the primary building block of FDDI rings. To manage the FDDI physical layer, a network manager must be able to add or remove FDDI stations from the rings via manipulation of the physical layer ports (PHY ports) of the wiring concentrator. In addition, greater visibility was given into the quality of the physical medium, using the link error monitor, for example.[1]

Another difference between the two technologies is the order of magnitude increase in performance of the FDDI ring over Ethernet/802.3. The ability to transparently connect 100-Mb FDDI token rings to 10-Mb Ethernet /802.3 LANs using the DECbridge 500 product greatly influenced network management requirements. Using these plug-and-play bridges makes it easier to create network topologies that can funnel high throughput FDDI traffic onto the lower bandwidth Ethernet/802.1 segments. A good management tool is required to monitor and control these topologies.

A final key difference is simply the need to manage the FDDI data link. The performance of the ring operation must be tuned, and having the capability to modify FDDI media access control (MAC) characteristics such as the valid transmission time view of the FDDI data link behavior.

Extended Management Capabilities

Beyond the technical requirements to provide FDDI network management, DECelms requirements were driven by the need for better network management capabilities in general. Providing an integrated product was a key project goal. Since managing FDDI products included managing bridges, the program team decided that the DECelms product would supersede the current bridge management product, remote bridge management software (RBMS), and incorporate RBMS functionality as a subset of the DECelms operating features. Thus, DECelms software needed to provide basic SET and SHOW capabilities for the LAN Bridge 100, LAN Bridge 150, and LAN Bridge 200 products, as well as for the new FDDI products, the DECbridge 500 and DECconcentrator 500 devices.

In addition to the ability to set and to show system parameters, DECelms software had requirements to provide automatic fault detection, automatic device discovery, performance monitoring, and FDDI ring mapping. Automatic or target token rotation time can provide the means to accomplish this tuning. Coupled with the modify operations is a rich set of SHOW capabilities, which gives a detailed

fault detection would be provided by a built-in polling mechanism and user alarms. Automatic device discovery would be accomplished by listening to system identification announcements broadcast by the devices and registering these devices in the DECelms registry. Performance statistics could be calculated from counters kept by the LAN Bridge 200 and DECbridge 500 products and presented to the user in tabular format. FDDI ring mapping would be accomplished by interpreting the data found in the SMT status information frame (SIF) configuration messages. These functions and their implementations are described in more detail later in this paper.

Development under Time-to-market Constraints

The requirements of the DECelms capabilities were stablizing. Program decisions were now influenced by the time-to-market constraints. Advanced development work on the DECbridge 500 and DECconcentrator 500 products was progressing, and this work was converted into a full-fledged product development effort. When the network management discussions began, this effort was well underway. Marketing and engineering management clearly communicated to the developers the expectation that Digital would be the first vendor months off the original 18-month schedule. The DECelms team was expected to meet these challenging time constraints.

Reducing the time spent on development required the DECelms team to make many trade-offs. The first trade-off concerned whether the product platform would comply with the Enterprise Management Architecture (EMA). The FDDI project was concurrent with DECmcc development (Digital's EMA-compliant director), but the FDDI program was three to six months ahead. Given the time constraints, the team decided that the FDDI program could not wait for the DECmcc program to catch up. A point product was the only solution that could be achieved in the short time frame. A second trade-off involved optimizing the development effort. Having no EMA experience, the team used the expertise it had recently gained from working on the RBMS version 2.0 development effort. They made the pragmatic decision to port the available RBMS code in order to meet the time-to-market constraints. The team planned to start with the RBMS code, delete code that was not applicable, and add code to provide the desired new functionality.

with a complete FDDI solution. This pressure prompted the developers to be very creative in choosing methodologies. As a result, the DECbridge 500 and DECconcentrator 500 teams shaved seven

But at the same time the RBMS code was being refined and expanded, the DECelms team had to keep the basic RBMS management code operational. The firmware teams needed to use DECelms code to debug their new code, so the DECelms code had to evolve in such a way that new code would replace old and supersede its functionality in a short period of time. This method of code development led to a series of operations akin to "brain transplants," where an interface was drawn in the existing code and new code containing additional functionality was added while the product was kept running. The timing of adding functions had to coincide with the development of the peer functionality in the firmware.

Network Management Architecture

With the product strategy taking shape, some difficult technical decisions concerning the management architecture had to be made. Choosing a network management protocol as well as supporting transport and network layer protocols was a major challenge.

Choosing a Protocol

The team identified three options for the management protocol implementation. The first option was using the common management information protocol (CMIP) layered on top of a pared-down implementation solely for implementing management agents in server products. This option was the most pure architectural solution and fit in well with Digital's long-term network strategy. But the effort was just beginning as an advanced development project, and the Phase V protocol specifications were still in the review process. The memory constraints imposed by the DECbridge 500 and DECconcentrator 500 devices coupled with the risk that the product might not meet the time-to-market constraints caused this option to be ruled out as a solution.

The second option was to use the CMIP protocol layered over a subset of the DNA Phase IV protocols. This subset is a streamlined implementation of the network services protocol (NSP) over an 802.3 data link. The network layer is null, so the protocol is limited to the extended LAN. This solution was small in size as dictated by the hardware, offered guaranteed end-to-end delivery service, and would take less time to implement than the DNA Phase V option; but there were several drawbacks. The bridge and wiring of the Digital Network Architecture (DNA) Phase V protocols. This implementation would include a subset of the DECnet Phase V session, transport, routing, and data link protocols used

concentrator management entities were not defined in terms of the EMA entity model, as was necessary to define the CMIP protocol structure. Using the Phase IV transport protocol option did not bring the management architecture closer to the open systems interconnect (OSI) model defined by DECnet/OSI Phase V, and, in fact, would result in "throwaway" transport and network layer code. Thus, this option was not suitable.

The third option was to extend the bridge management architecture and the RBMS protocol to include support for the FDDI products. In terms of pure architecture, this was the poorest solution, since it would merely result in the extension of an already limited protocol architecture. RBMS is layered over the Ethernet /802.3 data link, has a null network layer, and is therefore constrained to the extended LAN. The RBMS transport is connectionless and, thus, does not offer guaranteed end-to-end delivery service. In addition, no asynchronous message support exists, so the delivery of events or traps is unsupported. But RBMS is simple, being based on the IEEE 802.1 standard for network management, and easy to extend. Also, the LAN Bridge 200 development effort produced a new implementation of the management agent that could be ported into the DECbridge 500 and DECconcentrator 500 protocol was clearly the best possible choice.

SMT as Compared to Management over the Protocol Stack

Another approach to network management was via the FDDI station management frames defined in the American National Standards Institute (ANSI) X3T9.5 FDDI working group draft standard version 5.1. To provide this alternative meant adding complexity to SMT that belonged in a more robust management protocol such as CMIP. Further, the SMT standard was unstable in this area; many vendors participating in the standards work were advocating differing FDDI functionalities for SMT, thus trying to extend SMT beyond its originally intended scope. Digital's position was that network management should be performed using an OSI model where management is an application that runs on top of the protocol stack and is widely available. SMT frames are below the MAC level and, therefore, do not traverse a LAN beyond a local FDDI ring. Thus, management using SMT frames must come from a local station on the FDDI ring. The data provided by the SMT frames, such as the SIF, is valuable products. Further, there was the opportunity to port code from the RBMS version 2.0 product. Given the time constraints under which the DECelms team had to operate, the RBMS

to the network manager. The appropriate mechanism to communicate this data is a management protocol using an agent that can communicate across extended LANs. Once again, RBMS could perform this task.

Management Model Definition

After the protocol issue was settled, the next step was to define the FDDI manageable entities and attributes. The DECelms program team had decided to use RBMS, but the chosen long-term strategy was to use CMIP with the EMA guidelines. The RBMS decision affected only the server products and not the FDDI adapters that were also under development. The adapters would be managed via DECnet/OSI Phase V on both the VMS and the ULTRIX operating systems. The RBMS effort could not derail the long-term management strategy of EMA and the migration to OSI. Thus, there needed to be two management structures that offered similar management capabilities but used different mechanisms. Later, a third management structure driven by the internet community would become important, namely, the management information base (MIB) supported by the simple network management protocol (SNMP).

A series of FDDI network management meetings took place involving the FDDI implementors, the DNA architects, and members of the ANSI FDDI standards working group. The goal was to converge available. One source was the existing bridge management architecture, which defined how to manage an Ethernet/802.3 bridge such as the LAN Bridge 200 device. Another information source was the emerging ANSI X3T9.5 FDDI working group draft standard, specifically the chapter concerning FDDI station management. Additional information was found in an early draft of the DNA FDDI data link specification as well as in the approved version of the DNA CSMA/CD data link specification. The final set of inputs came from the product requirements of the DECbridge 500 and DECconcentrator 500 devices. These requirements called attention to management capabilities that went beyond bridging, wiring concentrators, FDDI data links, and Ethernet data links including management of the down-line-load upgrade feature and availability of field-replaceable unit (FRU) status.[2,3]

All of the above data sources had to be assimilated expeditiously because the chip designs were nearing final form. If the infrastructure necessary to allow for the on a set of manageable entities and attributes for the DECbridge 500 and DECconcentrator 500 products, while keeping an eye to the future of EMA. Several key sources of information were

extension of management functionality was not identified, either proposed features would be dropped from the products, or the chips would have to be respun. Keep in mind that the strategy was to have two management structures in place: the first, based on the bridge management architecture to be used for the DECbridge 500 and DECconcentrator 500 products and managed using the DECelms product; and, the second, incorporating the DNA FDDI data link architecture and a future bridge and concentrator management architecture based on the EMA. Design decisions regarding network management entities and attributes were made accordingly and are described below.

The following discussion presents the manner in which the combined model for the DECbridge 500 and DECconcentrator 500 products was developed. The proposed management model for the concentrator was flattened out and inserted into the current bridge architecture. The bridge architecture was extended to include the FDDI data link in addition to the Ethernet/802.3 data link. Additionally, the PHY port entity was added, but instead of being subordinate to the data link as in the management model for the bridge alone, the PHY port entity was at a peer level, as shown in Figure 1. Additional management formal DNA architecture, pending the results of the continuing ANSI standards meetings.

attributes were defined to bring the visibility into the box as required for each product. This phase of designing a network management architecture resulted in a delay in the development of the

Station Management Gateway

As mentioned earlier in this paper, SMT frames do not travel beyond a local FDDI ring. The DECelms product was expected to manage Digital and third-party FDDI stations on the FDDI ring using SMT frames. This management had to be initiated from the Ethernet /802.3 segment on which the DECelms host was located. The concept of the SMT gateway emerged from this situation.

An SMT gateway is firmware residing on the DECconcentrator 500 device (and subsequently on the DECbridge 500 product) that encapsulates SMT frames into RBMS protocol data units (PDUs) and forwards these frames across the extended LAN. The architecture that evolved to implement the SMT gateway was another extension of the RBMS protocol. A new entity type called the SMT gateway was added. Gateway requests and responses relating to SIFs, neighborhood information frames (NIFs), and echo frames were defined, for example, GET SIF, GET NIF, and DO ECHO. A set of timers was defined to allow the RBMS host, the SMT gateway, and the target station to be synchronized by datagram loss on the Ethernet. With the SMT gateway architecture in place, a DECelms management station located anywhere on the extended LAN can gather SMT frame data from any FDDI SMT version 5.1-compliant station on the same ring as a Digital SMT gateway station.

Management Beyond Set and Show

Several management functions beyond basic SET and SHOW capabilities were introduced in an earlier section, namely automatic fault detection, automatic device discovery, performance monitoring, and FDDI ring mapping. These features add value to the network manager. They present network fault and topology data in a timely and concise manner which frees the network manager from interpreting the microlevel details of the network. A primary goal of the DECelms team was to develop a set of tools that the network manager could use to detect and correct a network problem before it was reported by a distressed user.

Automatic Device Discovery

The addition of a station to an Ethernet/802.3 LAN is now a common and simple procedure. Using the and to avoid an ambiguous response. For example, if a response is missing, it is important to know if the lack of response is caused by gateway congestion, by no response from the target station, or simply

plug-and-play nature of Digital's bridge products, the connection of Ethernet /802.3 or FDDI LANs is also straightforward. The addition of stations to an FDDI ring is easy with the use of wiring concentrators. In campus environments, network managers can typically, but not always, control physical access to the network backbone but have less control once a segment enters an individual department. Consequently, there is a need to automatically discover the "renegade" devices that can be added to such networks. The DECelms product solves this problem by providing a device listener that listens to the maintenance operation protocol (MOP) system identification messages periodically broadcast by all of Digital's bridges and wiring concentrators. When a new station is heard, the DECelms software queries the station via the RBMS protocol to obtain more detailed information about the station and automatically adds the station to the DECelms registry. The software also produces a user alarm that notifies the network manager of the existence of this device and allows the manager to evaluate its impact on the topology and performance of the LAN. For instance, every bridge added between two stations adds latency to the protocol communications between the devices. Too much latency may negatively

The ability to quickly recognize changes in the state or counters of a network device can help a network manager avoid a degradation of service. The DECelms background poller provides this type of fault recognition by keeping information about the state of each device in the DECelms registry and reporting changes in that state to the network manager through user alarms and a log file. The information includes the designated bridge on a LAN, the number of FDDI ring initializations, the number of cyclic redundancy check (CRC) errors on a given data link, and, perhaps most importantly, the fact that a station has become unreachable.

Network Interface Multiplexer Process

Both the device listener and the background poller are integral parts of the DECelms process known as the network interface multiplexer (NIMUX). As implied by its name, NIMUX also provides the basic multiplexing of user protocol messages as they are sent and received by the data link driver. Designing NIMUX to incorporate these three distinct functions was a considerable challenge. The impact LAN performance by causing certain protocols to fail.

Background Poller

design includes a mailbox interface to the variable number of user interface tasks. This interface provides for both the sending and receiving of user data and alarms. The NIMUX design also provides the control interface for the background poller and the device listener and a service interface for the DECelms registry, which is only written by NIMUX for synchronization purposes. Within NIMUX is a kernel that synchronizes these three basic functions. Synchronization is

performed using VMS asynchronous system traps (ASTs) and event flags. When the NIMUX has no tasks, the process hibernates to conserve the CPU utilization. But, the NIMUX process may be woken up, for example, by the delivery of a mailbox message from a user process or by the completion of a network I/O operation.

The implementation of the background poller is complex. The background poller gathers state information for each device in the DECelms registry and continually circulates through the registry. This initial query is necessary to determine whether the

device is currently active or inactive. To obtain the complete state of a device, the DECelms software must issue multiple protocol requests to that device. In the case of a DECconcentrator 500 device, twelve requests must be itself. Several event flags are required to identify when a query is pending and then to identify whether the query is a success or a failure. When a single query is completed, the next step is dependent upon the device type and the status of the previous query.

The automatic addition of a device to the DECelms registry is performed in the polling cycle. Once a device MOP system identification message is heard and the device is recognized as new to DECelms, the responsibility of performing the RBMS protocol queries belongs to the poller. Three protocol messages are necessary to determine the device type and MAC addresses. The recognition that these two functions have this common thread helped to simplify the NIMUX design and limited the amount of internal state information that it was necessary to keep.

In parallel to the background device queries and the automatic device registration taking place, NIMUX also has to process user requests. These requests are given priority over those of the background poller, since issued to accumulate all the device data. The synchronization of these requests is a challenge, since it is possible for any of these requests to fail because of a datagram loss or the failure of the network or the device

an interactive user cannot be made to wait until all the background polling queries are completed. This processing is accomplished in the outermost loop of NIMUX. If a user request is delivered via a mailbox, an event flag indicates its arrival. This request is given preference over moving on to the next step in the polling process.

An important part of the interface between NIMUX and the user processes is the ability to deliver alarm messages. Each user process has the option of enabling or disabling alarm messages for its process. The fact that multiple alarms can be generated by a single device query from the background poller, and multiple users need to receive these alarms, means the interface has to contain queues to buffer the alarms and state variables to control the sending of the data. This problem was solved, as were most of the NIMUX interface problems, by the addition of these data structures to the bridge control process (BCP) tables. A BCP table is a robust data structure that keeps the NIMUX state information for a single user process. One table is created and destroyed for each user process that exists.

The last job NIMUX can perform is giving write access to the DECelms registry. Since both the automatic device registration function in NIMUX and a user process can add devices into the registry, a locking mechanism is needed to allow user registration, modification, and deletion of devices in the DECelms database using NIMUX as the server.

FDDI Ring Mapping

The ability to map the FDDI ring is a key feature of the DECelms product. The SMT gateway provides the mechanism to obtain the raw SMT frame data necessary to build ring maps. Two types of ring maps were identified as possible functionalities: a simple logical ring map and a physical ring map. The logical ring map could be built using the upstream neighbor address (UNA) contained in the NIFs. This functionality would provide a list of the MAC addresses in the FDDI ring to which the token was passed, but would not provide information regarding the number of PHY ports in a concentrator or the details of the physical topology of the network.

The preferred functionality was the physical ring map which can provide detail into the actual physical topology of the FDDI ring. Using the theory of wiring concentrators to build FDDI rings, the resulting topology is a ring of trees. Typically, this configuration includes dual registry, synchronization is necessary. The simplest solution is to eliminate the need to synchronize by having only NIMUX do the registry write operations. The mailbox interface commands were extended

attachment concentrators (DACs) located on the dual FDDI ring, with other concentrators and single attachment stations (SASs) connected to the concentrator on the dual ring in a tree-like fashion.[3] The physical ring map is required to represent this topology showing the PHY port attachments in addition to the MAC attachments.

This representation is accomplished using the data contained in the SIF configuration messages. The physical ring mapping starts when the user supplies the address of an SMT gateway on the target FDDI ring. The DECelms host then issues GET SIF gateway requests to the gateway. The algorithm calls for the ring map to start with the gateway's SIF information. Then, the UNA of the gateway contained in the SIF configuration message indicates which station in the ring to query next. Using the station descriptor, station state, and path descriptor fields of the SIF response, the DECelms host can derive the detailed physical map.

The basic mechanism to provide the physical ring map was understood, but the implementation and the actual application mandated additional requirements. With the SMT standard still in a state of flux, testing the ring map with third-party vendors disclosed that it was possible to interpret increased testing the ring map could interpret data from many vendors and thus became a very useful and popular tool.

Another feature was added to the ring map which made it possible to build partial ring maps. The capability of specifying the ring map starting address allows a user to pass a station that cannot be mapped otherwise (i.e., the station does not correctly respond to a SIF request) and manually continue the map with the next station on the ring.

the standard in different ways. Consequently, the ring map code had to be more flexible to interpret the same data fields which often had slight variations in format due to loose interpretations of the specification. With

The display of the map was a noteworthy task. Concepts for this implementation ranged from a simple tabular format to color graphics. The time constraints greatly influenced our decision. A simple and concise set of icons was developed to represent the stations on the ring. These icons are actually depicted as stick figures displayed using ASCII terminal art. Each icon is paired with the level at which it exists in a tree, similar to the way in which logic statements appear in a compiler listing. Figure 2 is an example of a ring map display. This icon implementation was achieved in far less time than a graphics display solution would have taken and offers the user a picture of the FDDI ring map, which is an advantage over simply presenting a table. In actual use, the map is easy to understand and provides an intuitive picture of the network topology.

User Interface

As described earlier, the DECelms code has its roots in the RBMS software. However, the RBMS user interface is a simple command line which lacks a number of features necessary to make the product more useful to network managers. To remedy this deficiency, the DECelms team set the following goals for the user interface. The software needed to provide intuitive commands to a previous RBMS user, shorter commands than RBMS, the ability to scroll through long output displays, the ability to input data from command files, and the ability to redirect output into files. During the process of reaching these goals, the product had to be continuously available as a development tool.

A screen-based, user interface solution, known as the DECelms screen manager (ESM), was designed. ESM uses VMS screen management (SMG) to provide the basic screen manipulation. Two screen modes were developed: an input mode and an output mode. In the input screen mode, commands are echoed and error messages displayed. keypad and options to move the output data into files. The integration of ESM into the DECelms software was accomplished through a well-defined interface that could take the output of the original RBMS output and convert it into the required format for ESM. One by one, the old screens were replaced by ESM, and at any given time during development there was a working user interface that provided debug support for the firmware developers.

Summary

This paper presents two important themes. The first concerns the technical challenges and accomplishments of the DECelms product. Among the challenges were defining the network management architecture, including the protocol, the manageable entities and attributes, and the SMT gateway. Technical accomplishments included the design of NIMUX and its multifaceted functionality. The second theme was how the development team proceeded to build the product. The time to market became the controlling factor in many product decisions. Trade-offs were made in favor of product functionality and

This mode provides command line recall, command line editing, and support for command files. The output screen mode is used to display the data from command requests. This mode provides the user with scrolling driven by the

meeting users' expectations rather than to promote and preserve architectural and design purity. Above all, product quality remained the top priority and the motivation for the DECelms development team to strive for excellence.

Acknowledgments

The DECelms product was a significant accomplishment, completed in a short period of time. It was the result of a great team effort and personal sacrifice by the DECelms development team. I want to recognize the following individuals for their efforts as members of the engineering team: Barry Colella, Mary Ellen Monette, Phil Morano, Ken Pruyn, and Herman Levenson. In addition, I want to thank Andrew Shores for his diligence and endurance in producing the DECelms user documentation. Lastly, I want to thank members of the FDDI program team, the DECbridge 500 and DECconcentrator 500 teams, and the NAC architecture group for their support and contributions that made the DECelms product a success story.

References

o P. Ciarfella, D. Benson, and D. Sawyer, "An Overview of the Common Node Software," Digital Technical Journal, vol. 3, no. 2 (Spring 1991, this issue): 42-52.

o R. Kochem, J. Hiscock, and B. Mayo, "The Development of the DECbridge Product," Digital Technical Journal, vol. 3, no. 2 (Spring 1991, this issue): 53-63.

o W. Tiffany, P. Koning, and J. Kuenzel, "The DECconcentrator 500 Product," Digital Technical Journal, vol. 3, no. 2 (Spring 1991, this issue): 64-75.