A DNP3 Protocol Primer

This is a primer for people who want a quick understanding of DNP3 without having to comb through the tedious details of a complex specification.

So let us start with what it is. Protocols define the rules by which devices talk with each other, and DNP3 is a protocol for transmission of data from point A to point B using serial communications. It has been used primarily by utilities like the electric companies, but it operates suitably in other areas.

A typical electric company may have a centralized operations center that monitors the state of all the equipment in each of its substations. In the operations center, a powerful computer stores all of the incoming data and displays the system for the human operators. Substations have many devices that need monitoring (are circuit breakers opened or closed?), current sensors (how much current is flowing?) and voltage transducers (what is the line potential?). That only scratches the surface; a utility is interested in monitoring many parameters, too numerous to discuss here. The operations personnel often need to switch sections of the power grid into or out of service. One or more computers are situated in the substation to collect the data for transmission to the master station in the operations center. The substation computers are also called upon to energize or de-energize the breakers and voltage regulators.

DNP3 provides the rules for substation computers and master station computers to communicate data and control commands. DNP3 is a non-proprietary protocol that is available to anyone. Only a nominal fee is charged for documentation, but otherwise it is available worldwide with no restrictions. This means a utility can purchase master station and substation computing equipment from any manufacturer and be assured that they will reliably talk to each other. Vendors compete based upon their computer equipment's features, costs and quality factors instead of who has the best protocol. Utilities are not stuck with one manufacturer after the initial sale.

What do the computers talk about? The substation computer gathers data for transmission to the master as

1. Binary input data that is useful to monitor two-state devices. For example a circuit breaker is closed or tripped or a pipeline pressure alarm shows normal or excessive.
2. Analog input data that conveys voltages, currents, power, reservoir water levels and temperatures.
3. Count input data that reports kilowatt hours of energy.
4. Files that contain configuration data.

The master station issues control commands that take the form of

1. Close or trip a circuit breaker, raise or lower a gate, and open or close a valve.
2. Analog output values to set a regulated pressure or set a desired voltage level.

Other things the computers talk to each other about are synchronizing the time and date, sending historical or logged data, waveform data, and on and on.

DNP3 was designed to optimize the transmission of data acquisition information and control commands from one computer to another. It is not a general purpose protocol for transmitting hypertext, multimedia or huge files.

The terms server and client are applicable to DNP3 systems. For our purposes, the definition of a server is a device or software process that has data or information that someone else wants. Substation computers are servers. A client is a device or software process that requests data from a server. A master station is a client.

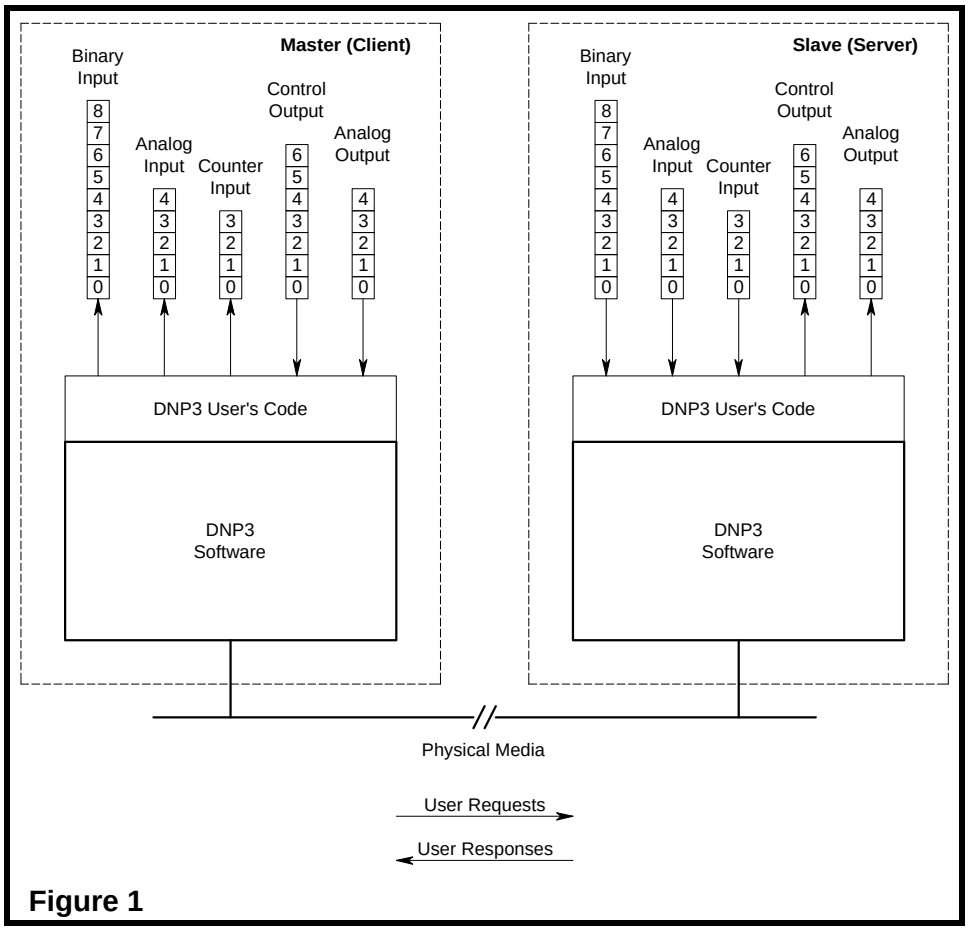

Figure 1

Figure 1 shows the client-server relationship and gives a simplistic view of the databases and software processes involved. The master or client is on the left side of figure 1, and the slave or server is on the right side.

A series of square blocks at the top of the server depicts its databases and output devices. The various data types are conceptually organized as arrays. An array of binary input values represents states of physical or logical boolean devices. Values in the analog input array represent input quantities that the server measured or computed. An array of counters represents count values, such as kilowatt hours, that are ever increasing (until they reach a maximum and then roll over to zero and start counting again.) Control outputs are organized into an array representing physical or logical on-off, raise-lower and trip-close points. Lastly, the array of analog outputs represents physical or logical analog quantities such as those used for setpoints.

The elements of the arrays are labeled 0 through N - 1 where N is the number of blocks shown for the respective data type. In DNP3 terminology, the element numbers are called the point indexes. Indexes are zero-based in DNP3, that is, the lowest element is always identified as zero. Some protocols use 1-based indexing.

Notice that the DNP3 client, or master, also has a similar database for the input data types (binary, analog and counter.) The master, or client, uses values in its database for the specific purposes of displaying system states, closed-loop control, alarm notification, billing, and much, much more. An objective of the client is to keep its database updated. It accomplishes this by sending requests to the server (slave) asking it to return the values in the server's database. This is termed polling. The server responds to the client's request by transmitting the contents of its database. Arrows are drawn at the bottom of figure 1 showing the direction of the requests (toward the server) and the direction of the responses (toward the client.) Later we will discuss systems whereby the slaves transmit responses without being asked.

The client and the server shown in figure 1 each have two software layers. The top layer is the DNP3 user layer. In the client, it is the software that interacts between the database and initiates the requests for the server's data. In the server, it is the software that fetches the requested data from the server's database for responding to client requests. It is interesting to note, that if no physical separation of the client and server existed, eliminating the DNP3 might be possible by connecting these two upper layers together. However, since physical, or possibly logical separation of the client and server exists, DNP3 software is placed at a lower level. The DNP3 user's code uses the DNP3 software for transmission of requests or responses to the matching DNP3 user's code at the other end.

More will be said about data types and software layers later, but first we want to examine a few typical system architectures where DNP3 is used.

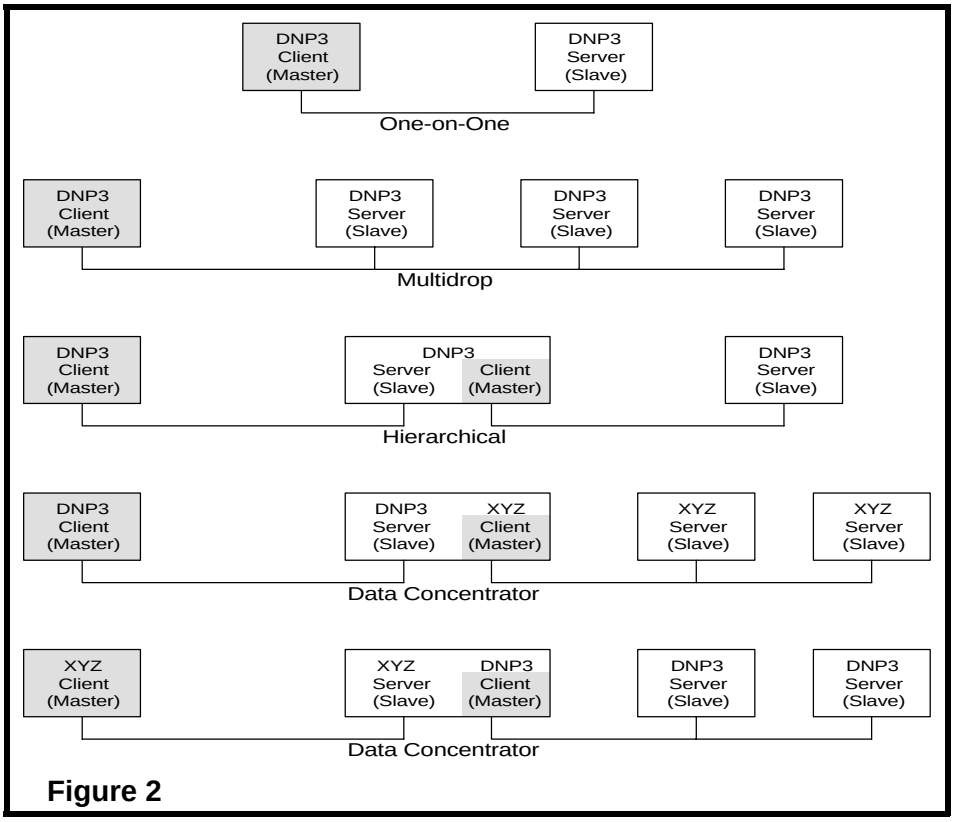

Figure 2

Figure 2 shows common system architectures in use today. At the top is a simple one-on-one system having one master station and one slave. The physical connection between the two is typically a dedicated or dial-up telephone line.

The second type of system is known as a multidrop design. One master station communicates with multiple slave devices. Conversations are typically between the client and one server at a time. The master requests data from the first slave, then moves onto the next slave for its data, and continually interrogates each slave in a round robin order. The communication media is a multi-dropped telephone line, fiber optic cable, or radio. Each slave can hear messages from the master and is only permitted to respond to messages addressed to itself. Slaves may or may not be able to hear each other.

In some multidrop forms, communications are peer-to-peer. A station may operate as a client for gathering information or sending commands to the server in another station. And then, it may change roles to become a server to another station.

The middle row in figure 2 shows a hierarchical type system where the device in the middle is a server to the client at the left and is a client with respect to the server on the right. The middle device is often termed a sub-master.

Both lines at the bottom of figure 2 show data concentrator applications and protocol converters. A device may gather data from multiple servers on the right side of the figure and store this data in its database where it is retrievable by a master station client on the left side of the figure. This design is often seen in substations where the data concentrator collects information from local intelligent devices for transmission to the master station.

In recent years, several vendors have used TCP/IP to transport DNP3 messages in lieu of the media discussed above. Link layer frames, which we have not talked about yet, are embedded into TCP/IP packets. This approach has enabled DNP3 to take advantage of Internet technology and permitted economical data collection and control between widely separated devices.

Many communication circuits between the devices are imperfect. They are susceptible to noise and signal distortion.

The DNP3 software is layered to provide reliable data transmission and to effect an organized approach to the transmission of data and commands. Figure 3 shows the layering that was not shown in figure 1.

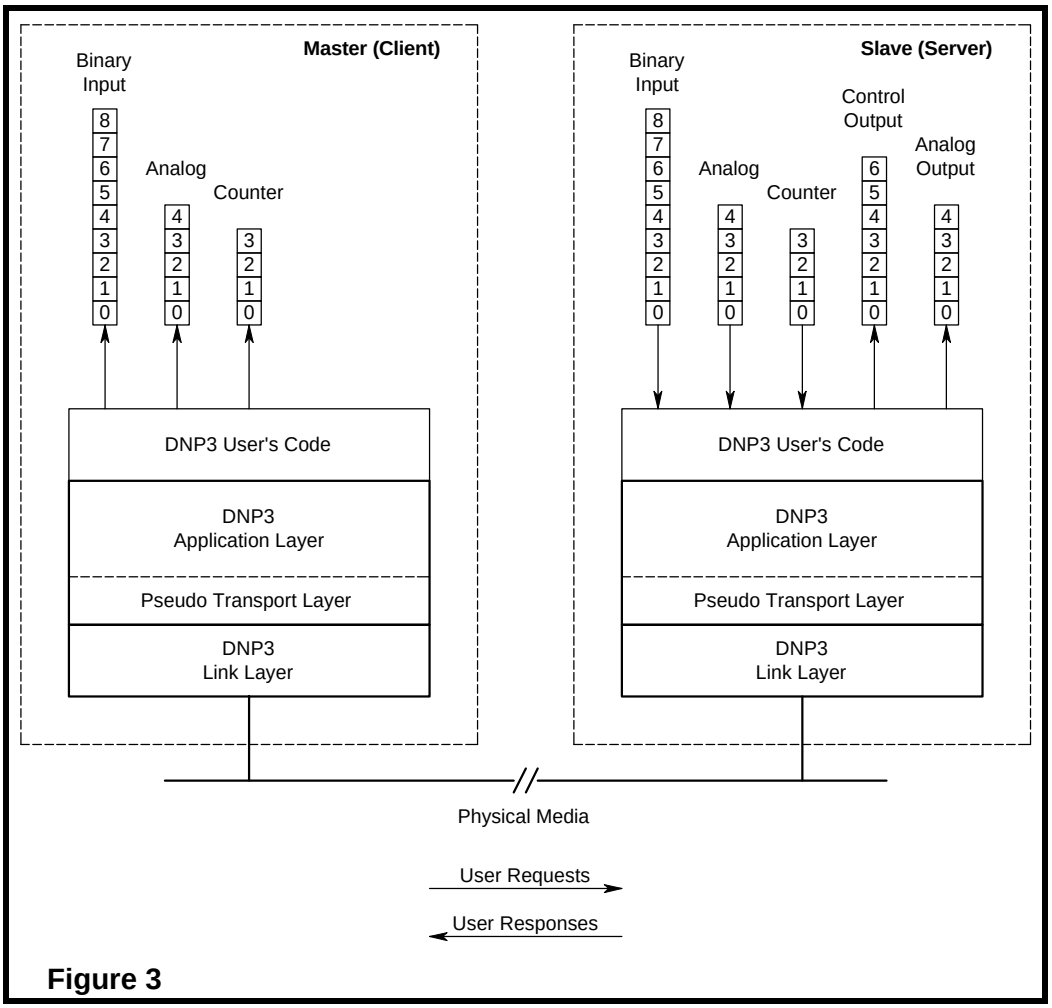

Figure 3

The link layer has the responsibility of making the physical link reliable. It does this by providing error detection and duplicate frame detection. The link layer sends and receives packets, which in DNP3 terminology, are called frames.

Sometimes transmission of more than one frame is necessary to transport all of the information from one device to another.

A DNP3 frame consists of a header and data section. The header specifies the frame size, which DNP3 station should receive the frame, which DNP3 device sent the frame and data link control information. The data section is commonly called the payload and contains the data passed down from the layers above.

DNP3 Frame

Header	Data

Header

Sync	Length	Link Control	Destination Address	Source Address	CRC

Every frame begins with two sync bytes that help the receivers determine where the frame begins. The length specifies the number of octets in the remainder of the frame, not including CRC check octets. The link control octet is used between sending and receiving link layers to coordinate their activities.

A destination address specifies which DNP3 device should process the data, and the source address identifies which DNP3 device sent the message. Having both destination and source addresses satisfies at least one requirement for peer-to-peer communications because the receiver knows where to direct its responses. 65520 individual addresses are available. Every DNP3 device must have a unique address within the collection of devices sending and receiving messages to and from each other. Three destination addresses are reserved by DNP3 to denote an all-call message; that is, the frame should be processed by all DNP3 devices. Thirteen addresses are reserved for special needs in the future.

The data payload in the link frame contains a pair of CRC octets for every 16 data octets. This provides a high degree of assurance that communication errors can be detected. The maximum number of octets in the data payload is 250, not including CRC octets. (The longest link layer frame is 292 octets if all the CRC and header octets are counted.)

One often hears the term “link layer confirmation” when DNP3 is discussed. A feature of DNP3's link layer is the ability for the transmitter of the frame to request the receiver to confirm that the frame arrived. Using this feature is optional, and it is often not employed. It provides an extra degree of assurance of reliable communications. If a confirmation is not received, the link layer may retry the transmission. Some disadvantages are the extra time required for confirmation messages and waiting for multiple timeouts when retries are configured.

It is the responsibility of the transport layer to break long messages into smaller frames sized for the link layer to transmit, or when receiving, to reassemble frames into the longer messages. In DNP3 the transport layer is incorporated into the application layer. The transport layer requires only a single octet within the message to do its work. Therefore, since the link layer can handle only 250 data octets, and one of those is used for the transport function, then each link layer frame can hold as many as 249 application layer octets.

Application layer messages are broken into fragments. Fragment size is determined by the size of the receiving device’s buffer. It normally falls between 2048 and 4096 bytes. A message that is larger than a one fragment requires multiple fragments. Fragmenting messages is the responsibility of the application layer.

Note that an application layer fragment of size 2048 must be broken into 9 frames by the transport layer, and a fragment size of 4096 needs 17 frames. Interestingly, it has been learned by experience that communications are sometimes more successful for systems operating in high noise environments if the fragment size is significantly reduced.

The application layer works together with the transport and link layers to enable reliable communications. It provides standardized functions and data formatting with which the user layer above can interact. Before functions, data objects and variations can be discussed, the terms static, events and classes need to be covered.

In DNP3, the term static is used with data and refers to the current value. Thus static binary input data refers to the present on or off state of a bi-state device. Static analog input data contains the value of an analog at the instant it is transmitted. One possibility DNP3 allows is requesting some or all of the static data in a slave device.

DNP3 events are associated with something significant happening. Examples are state changes, values exceeding some threshold, snapshots of varying data, transient data and newly available information. An event occurs when a binary input changes from an on to an off state or when an analog value changes by more than its configured deadband limit. DNP3 provides the ability to report events with and without time stamps so that the client can generate a time sequence report.

The user layer can direct DNP3 to request events. Usually, a client is updated more rapidly if it mostly polls for events from the server and only occasionally asks for static data as an integrity measure. The reason updates are faster is because the number of events generated between server interrogations is small and, therefore, less data must be returned to the client.

DNP3 goes a step further by classifying events into three classes. When DNP3 was conceived, class 1 events were considered as having higher priority than class 2 events, and class 2 were higher than class 3 events. While that scheme can be still be configured, some DNP3 users have developed other strategies more favorable to their operation for assigning events into the classes. The user layer can request the application layer to poll for class 1, 2 or 3 events or any combination of them.

DNP3 has provisions for representing data in different formats. Examination of analog data formats is helpful to understand the flexibility of DNP3. Static, current value, analog data can be represented by variation numbers as follows:

1. A 32-bit integer value with flag,
2. A 16-bit integer value with flag,
3. A 32-bit integer value,
4. A 16-bit integer value,
5. A 32-bit floating point value with flag and
6. A 64-bit floating point value with flag.

   The flag referred to is a single octet with bit fields indicating whether the source is on-line, value contains a restart value, communications are lost with the source, the data is forced and the value is over range.

Not all DNP3 devices can transmit or interpret all six variations. Later, DNP3 levels are discussed, but for now, suffice it to say that DNP3 devices must be able to transmit the simplest variations so that any receiver can interpret the contents.

Event analog data can be represented by these variations:

1. A 32-bit integer value with flag,
2. A 16-bit integer value with flag,
3. A 32-bit integer value with flag and event time,
4. A 16-bit integer value with flag and event time,
5. A 32-bit floating point value with flag,
6. A 64-bit floating point value with flag,
7. A 32-bit floating point value with flag and event time and
8. A 32-bit floating point value with flag and event time.

   The flag has the same bit fields as for the static variations.

It looks like a variation one or two analog event cannot be differentiated from a variation one or two static analog value. DNP3 solves this predicament by assigning object numbers. Static analog values are assigned as object 30, and event analog values are assigned as object 32. Static analog values, object 30, can be formatted in one of 6 variations, and event analog values, object 32, can be formatted in one of 8 variations.

When a DNP3 server transmits a message containing response data, the message identifies the object number and variation of every value within the message. Object and variation numbers are also assigned for counters, binary inputs, controls and analog outputs. In fact, all valid data types and formats in DNP3 are identified by object and variation numbers. Defining the allowable objects and variations helps DNP3 assure interoperability between devices. DNP3's basic documentation contains a library of valid objects and their variations.

The client's user layer formulates its request for data from the server by telling the application layer what function to perform, like reading, and specifying which objects it wants from the server. The request can specify how many objects it wants or it can specify specific objects or a range of objects from index number X through index number Y. The application layer then passes the request down through the transport layer to the link layer that, in turn, sends the message to the server. The link layer at the server checks the frames for errors and passes them up to the transport layer where the complete message is assembled in the server's application layer. The application layer then tells the user layer which objects and variations were requested.

Responses work similarly, in that, the server's user layer fetches the desired data and presents it to the application layer that formats the data into objects and variations. Data is then passed downward, across the communication channel and upward to the client's application layer. Here the data objects are presented to the user layer in a form that is native to the client's database.

Reading data was mentioned in the above two paragraphs, but DNP3 software is designed to handle other functions. For one the client can set the time in the server. The client can transmit freeze accumulator requests, and it can transmit requests for control operations and setting of analog output values using select-before-operate or direct-operate sequences.

One area that has not been covered yet is transmission of unsolicited messages. This is a mode of operating where the server spontaneously transmits a response, possibly containing data, without having received a specific request for the data. Not all servers have this capability, but those that do must be configured to operate in this mode. This mode is useful when the system has many slaves and the master requires notification as soon as possible after a change occurs. Rather than waiting for a master station polling cycle to get around to it, the slave simply transmits the change.

To configure a system for unsolicited messages, a few basics need to be considered. First, spontaneous transmissions should generally occur infrequently, otherwise, too much contention can occur, and controlling media access via master station polling would be better. The second basic issue is that the server should have some way of knowing whether it can transmit without stepping on someone else's message in progress. DNP3 leaves specification of algorithms to the system implementor.

One last area of discussion involves implementation levels. The DNP3 organization recognizes that supporting every feature of DNP3 is not necessary for every device. Some devices are limited in memory and speed and do not need specific features, while other devices must have the more advanced features to accomplish their task. DNP3 organizes complexity into three levels. At the lowest level, level 1, only very basic functions must be provided and all others are optional. Level 2 handles more functions, objects and variations, and level 3 is even more sophisticated. Within each level only certain combinations of request formats and response formats are required. This was done to limit software code in clients and servers while still assuring interoperability.

It should be apparent by now that DNP3 is a protocol that fits well into the data acquisition world. It transports data as generic values, it has a rich set of functions, and it was designed to work in a wide area communications network. The standardized approach of objects and variations, and link, transport and application layers, plus public availability makes DNP3 a protocol to be regarded.

Author:

Ken Curtis from Woodland Engineering wrote this paper to help the many people who are just getting into or considering DNP3 for their operation. Ken is a consulting engineer that has been contracted to write software for DAQ Electronics who also sponsors his participation in the DNP Technical Committee. Valuable editing assistance was provided by Mike Thesing of Advanced Control Systems.

DNP Users Group:

Mail Address: DNP Users Group
PO Box 43075, DVPO
Calgary, AB T2J 7A7
Canada

Fax: 403-271-1319

Email: dnp@home.com

Website: www.dnp.org

