A Sierra Monitor Company

Driver Manual

(Supplement to the FieldServer Instruction Manual)

FS-8704-15 HTTP XML Driver

APPLICABILITY & EFFECTIVITY

Effective for all systems manufactured after August 2012

Driver version:	1.11
Document Revision:	2

TABLE OF CONTENTS

1 HTTP XML DRIVER DESCRIPTION

The *Ethernet HTTP XML Driver* driver allows the FieldServer to transfer data to and from devices over Ethernet using HTTP/XML Driver protocol. The FieldServer can emulate either a Server or Client.

The XML Driver is built on HTTP web technology (Port 80) and it uses pages formatted in XML syntax to respond to HTTP requests or allows XML responses to be decoded and stored. Both a client and a server are supported.

The Server side is an XML formatted response of the internal Data Array structure contained within the FieldServer, requested from a Remote Client device to the FieldServer URL.

The Client uses a HTTP GET request to a specified URL to request XML data. The driver has the ability to decode the XML response and store different Elements uniquely identified by some attribute within the Element. The data of the matching Element is stored in the FieldServer Data Arrays.

FieldServer Mode	Comments
Server	This mode is always enabled within the XML driver, and is requested by "http://<ip address>/data_arrays.xml" where <ip address> corresponds to the FieldServer IP address.
Client	Supports multiple client connections to different URLs, with associated decoding map descriptors linked to an active GET URL request.

Max Nodes Supported

FieldServer Mode	Nodes	Comments
Client	32	The Driver can establish a connection to several different IP addresses
Server	0	The Server is always enabled and defined on the connection, hence the driver does not allow any server node selection

2 DRIVER SCOPE OF SUPPLY

2.1 Supplied by FieldServer Technologies for this driver

FieldServer Technologies PART #	Description
FS-8915-10	UTP cable (7 foot) for Ethernet connection

2.2 Provided by the Supplier of 3rd Party Equipment

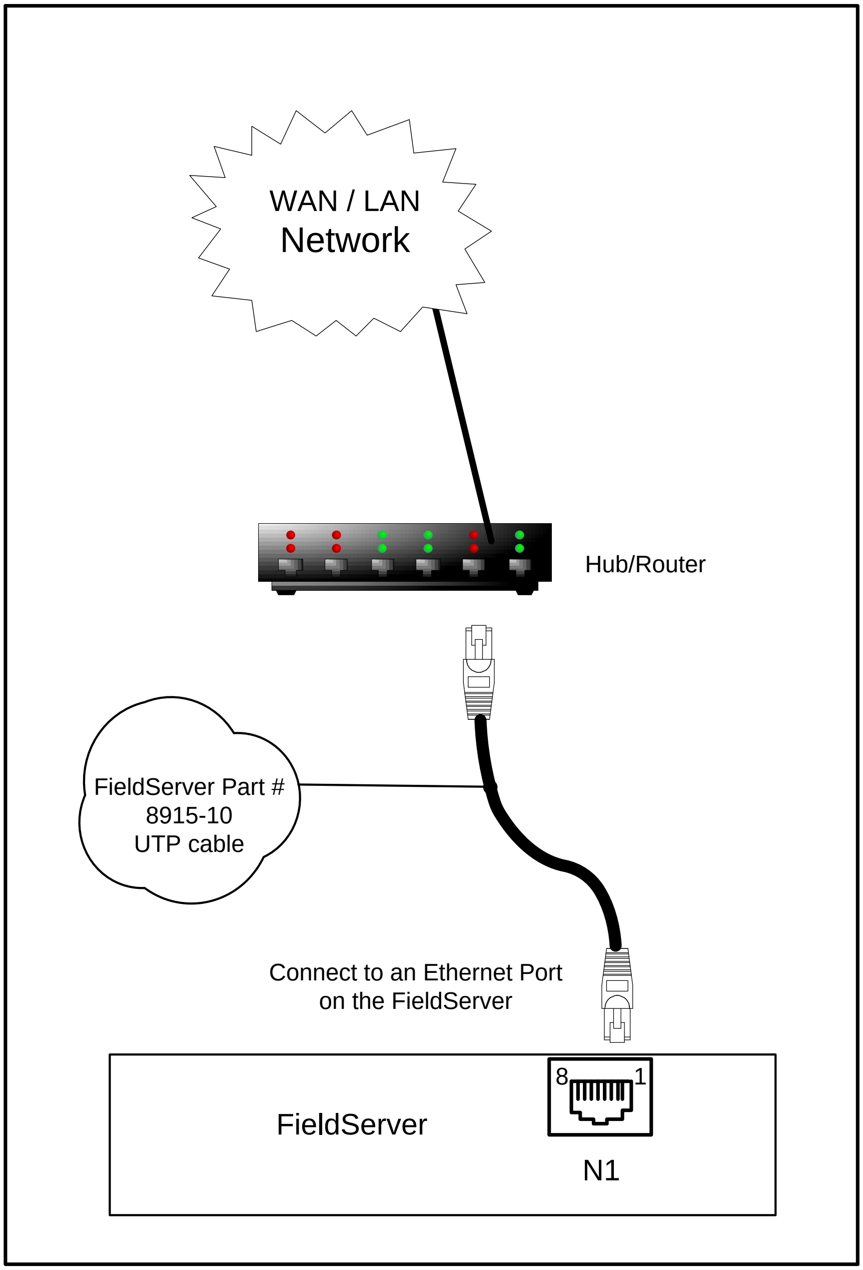
2.2.1 Required 3rd Party Hardware

Part #	Description
	Ethernet 10/100 BaseT hub*

* Not all FieldServer models support 100BaseT. Consult the appropriate instruction manual for details of the Ethernet speed supported by specific hardware.

3 HARDWARE CONNECTIONS

The FieldServer is connected to the Network as shown in connection drawing.



The Ethernet connection can be achieved using a switch or crossover cable. The driver will support all Ethernet connections on a local area network (LAN) or wide area network (WAN) including internet connections and crossover cable connections.

4 DATA ARRAY PARAMETERS

Data Arrays are “protocol neutral” data buffers for storage of data to be passed between protocols. It is necessary to declare the data format of each of the Data Arrays to facilitate correct storage of the relevant data.

Section Title		
Data_Arrays		
Column Title	Function	Legal Values
Data_Array_Name	Provide name for Data Array	Up to 15 alphanumeric characters
Data_Array_Format	Provide data format. Each Data Array can only take on one format.	Float, Bit, Uint16, Sint16, Uint32, Sint32, Byte.
Data_Array_Length	Number of Data Objects. Must be larger than the data storage area required by the Map Descriptors for the data being placed in this array.	1-10, 000

Example

```
//    Data Arrays
Data_Arrays
Data_Array_Name     , Data_Array_Format     , Data_Array_Length
DA_AI_01            , Float                 , 200
DA_AO_01            , Float                 , 200
DA_DI_01            , Bit                   , 200
DA_DO_01            , Bit                   , 200
```

5 CONFIGURING THE FIELDSERVER AS A HTTP XML DRIVER CLIENT

For a detailed discussion on FieldServer configuration, please refer to the FieldServer Configuration Manual. The information that follows describes how to expand upon the factory defaults provided in the configuration files included with the FieldServer (See ".csv" sample files provided with the FieldServer).

This section documents and describes the parameters necessary for configuring the FieldServer to communicate with an XML Server running over a web server on port 80.

The configuration file tells the FieldServer about its target XML data, and the decoding of data required. In order to enable the FieldServer as an XML Client, the driver independent FieldServer buffers need to be declared in the "Data Arrays" section, the destination XML Servers IP addresses need to be declared in the "Client Side Nodes" section, and the data required from the XML requests needs to be mapped in the "Client Side Map Descriptors" section. Details on how to do this can be found below.

Note that in the tables, * indicates an optional parameter, with the bold legal value being the default.

5.1 Client Side Connection Parameters

Section Title		
Connections		
Column Title	Function	Legal Values
Connection	Connection Name	N1, N2[1]
Protocol	Specify protocol used	XML-HTTP, HTTP, XML

Example

```
//   Client Side Connections

Connections
Connection          , Protocol
N1                  , XML
```

5.2 Client Side Node Parameters

Section Title		
Nodes		
Column Title	**Function**	**Legal Values**
Node_Name	Provide name for Node	Up to 32 alphanumeric characters
IP_Address	Destination IP Address	xxx.xxx.xxx.xxx
Protocol	Specify Protocol used	XML-HTTP, HTTP, XML
Connection	Specify which network adapter to use.	N1, N2[2]

[1] Not all ports shown are necessarily supported by the hardware. Consult the appropriate Instruction manual for details of the ports available on specific hardware.

Example

```
//    Client Side Nodes

Nodes
Node_Name    , IP_Address      , Protocol     , Connection
FS_WEB_01    , 192.168.1.10    , XML          , N1
```

5.3 Client Side Map Descriptor Parameters

5.3.1 FieldServer Specific Map Descriptor Parameters

Column Title	Function	Legal Values
Map_Descriptor_Name	Name of this Map Descriptor	Up to 32 alphanumeric characters
Data_Array_Name	Name of Data Array where data is to be stored in the FieldServer	One of the Data Array names from Section 0.
Data_Array_Offset	Starting location in Data Array	0 to (Data_Array_Length-1) as specified in Section 0.
Function	Function of Client Map Descriptor	Rdbc, Wrbc, Wrbx, Passive_Client

5.3.2 Driver Related Map Descriptor Parameters

Column Title	Function	Legal Values
Node_Name	Name of Node to fetch data from	One of the Node names specified in Section 5.2
Length	Length of Map Descriptor, Number of Elements to store into data arrays, this needs to be separated by some non-numeric character, e.g. space or comma	
Method*	HTTP Fetch Method for active Map descriptors	**GET** , POST
XML-URL	HTTP URL, this does not include the IP	Up to 200 alphanumeric characters (special parameters outlined in 5.3.5 allowed)
Write_Command*	When using POST method this is the payload of the data sent to the address	Up to 200 alphanumeric characters (special parameters outlined in 5.3.4 allowed)
Linked_Map_Descriptor	This is the active mapdescriptor responsible for fetching the XML data	Up to 32 alphanumeric characters
Element*	XML Element to store from the XML response. The syntax is as follows “***Elment.Child Element. Grand Child Element***.**Attribute**” If the attribute is left out the driver store the Element contents. If left out the driver store the Data of the furtherest childs data of the matching element	Up to 200 alphanumeric characters

Search_Value*	String Search value to uniquely match the Element the driver want to store, this could be any Attribute of the XML response. If left out the driver use the first element to store the data.	Up to 200 alphanumeric characters
Search_Attribute*	Used to specify which Attribute the driver want the search value to match.	Up to 200 alphanumeric characters
HTTP_RESP_DA	Used for Writes where the driver want to store the HTML response from the server for diagnostics.	Up to 32 alphanumeric characters

5.3.3 Timing Parameters

Column Title	Function	Legal Values
Scan_Interval	Rate at which data is polled	≥0.001s

5.3.4 Special Keywords for URL and Write_Command

Parameter	Description
<DATA_ARRAY_NAME.OFFSET>	Substitute entry with value from matching data array and offset
<local>	Substitute value with Map descriptors first data array entry, or data offset from match cache block write
<offset>	Used for passive map descriptors to indicate which offset changed when the cache block gets created.

5.4 Map Descriptor Example.

```
//    Client Side Map Descriptors

Map_Descriptors
Map_Descriptor_Name   , Function         , Data_Array_Name  , Data_Array_Index  , Node_Name         , Length   , Scan_Interval   , XML-URL          , Method
CMD_DA_GET1           , RDBC             , DA_HTTP_STR1     , 0                 , FS_WEB_01         , 500      , 0               , DATA_ARRAYS.XML  , GET

Map_Descriptors
Map_Descriptor_Name   , Function         , Data_Array_Name  , Data_Array_Index  , Node_Name         , Length   , Linked_Map_Descriptor
CMD_RD_UINT16         , PASSIVE_CLIENT   , DA_UINT16        , 0                 , FS_WEB_01         , 100      ,CMD_DA_GET1

...                   , Element                             , Search_Value      , Search_Attribute  , Method   , XML-URL         , Write_Command
...                   , DATA_ARRAYS.DATA_ARRAY.DATA         , DA_AI_01          , name              , POST     , post.cgi        , DA_UINT16.<offset>=<local>

// Write section
Map_Descriptors
Map_Descriptor_Name   , Function         , Data_Array_Name  , Data_Array_Index  , Node_Name         , Length   , Scan_Interval
CMD_WR_UINT16a        , WRBC             , DA_UINT16_WR     , 0                 , FS_WEB_01         , 100      , 2s

...                   ,  Method          , XML-URL          , Write_Command
...                   ,  POST            , post.cgi         , "DA_UINT16.0=<DA_UINT16_WR.0>&DA_UINT16.1=<DA_UINT16_WR.1>"
```

XML Element to store, and Search Value to match on, and which XML attribute to match

URL and payload of a regular POST command generated by the FieldServer

If the data changes, do a POST to the following URL with the payload

Linked Map Descriptors, these are where the XML request gets stored too.

Primary read map descriptor with URL. And method is a GET. HTTP response is stored to the data array. This is mainly for diagnostic purposes.

6 CONFIGURING THE FIELDSERVER AS A HTTP XML DRIVER SERVER

For a detailed discussion on FieldServer configuration, please refer to the FieldServer Configuration Manual. The information that follows describes how to expand upon the factory defaults provided in the configuration files included with the FieldServer (See ".csv" sample files provided with the FieldServer).

This section documents and describes the parameters necessary for configuring the FieldServer to communicate with an XML Client. This can be any web browser such as internet explorer or Firefox. Or could be any other XML client application

Note that in the tables, * indicates an optional parameter, with the bold legal value being the default.

6.1 Server Side Connection Parameters

Section Title		
Connections		
Column Title	**Function**	**Legal Values**
Connection	Connection Name	N1 / N2
Protocol	Specify protocol used	**XML-HTTP**, HTTP, XML

Example

```
//   Server Side Connections
Connections
Connection          , Protocol
N1                  , XML
```

6.2 Server Side Node Parameters

Nodes do not apply to the server side of this protocol

6.3 Server Side Map Descriptor Parameters

Map descriptors do not apply to the server side of this protocol

6.4 Server side requests and Responses supported

Request:

```
http://192.168.1.5/data_arrays.xml
```

Response:

```
<DATA_ARRAYS FST_XML_VERSION="1.00" MAX_INDEX="5" BRIDGE_TITLE="Lonworks Server">
        <DATA_ARRAY NAME="DA_Group" FORMAT="Byte" LENGTH="99" INDEX="1">
                <DATA OFFSET="0" DATA_AGE="3:00:39:36.736s" STATUS="0">0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
        0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
        0 0 0 0 0 0 0 0 0 0 0 0 0 0</DATA>
        </DATA_ARRAY>
        <DATA_ARRAY NAME="DA_P16_1Sta" FORMAT="Byte" LENGTH="2" INDEX="2">
                <DATA OFFSET="0" DATA_AGE="34.705s" STATUS="0">0 0</DATA>
        </DATA_ARRAY>
        <DATA_ARRAY NAME="DA_Pnl16_1" FORMAT="Byte" LENGTH="8" INDEX="3">
                <DATA OFFSET="0" DATA_AGE="3:00:39:36.646s" STATUS="0">0 0 0 1 0 0 0 0</DATA>
        </DATA_ARRAY>
        <DATA_ARRAY NAME="DA_P16_2Sta" FORMAT="Byte" LENGTH="2" INDEX="4">
                <DATA OFFSET="0" DATA_AGE="34.050s" STATUS="0">0 0</DATA>
        </DATA_ARRAY>
        <DATA_ARRAY NAME="DA_Pnl16_2" FORMAT="Byte" LENGTH="8" INDEX="5">
                <DATA OFFSET="0" DATA_AGE="3:00:39:36.548s" STATUS="0">1 0 0 0 0 0 0 0</DATA>
        </DATA_ARRAY>
</DATA_ARRAYS>
```

Section Title		
Optional Selective Data_Array.xml parameters		
	Function	**Legal Values**
?name=DA_NAME	Specify which Data array you want to retrieve	Up to 15 alphanumeric characters
:LENGTH	Number of values to return	**>=1**
.OFFSET	Start at which offset in the Data Array	**>=0**
&name= …	Add another Data Array to the response	Same as above.

Example

```
http://192.168.1.5/data_arrays.xml?name=DA_REG01.01:5&name=DA_REG02.01:5
```

Appendix A. XML Server Schema

Below find the XML Schema used for Server responses to the data_Arrays.xml request.

```
<xs:schema elementFormDefault="qualified" xmlns:xs="http://www.w3.org/2001/XMLSchema">
 <xs:element name="DATA_ARRYS">
   <xs:attribute name="FST_XML_VERSION" type="xs:string" />
   <xs:attribute name="MAX_INDEX" type="xs:string" />
   <xs:attribute name="BRIDGE_TITLE" type="xs:string" />
   <xs:complexType>
     <xs:sequence>
       <xs:element name="DATA_ARAY">
         <xs:attribute name="NAME" type="xs:string" />
              <xs:attribute name="FORMAT" type="xs:string" />
              <xs:attribute name="LENGTH" type="xs:string" />
              <xs:attribute name="INDEX" type="xs:string" />
              <xs:complexType>
                <xs:element name="Data" type="xs:string">
                  <xs:attribute name="OFFSET" type="xs:string" />
                  <xs:attribute name="DATA_AGE" type="xs:string" />
                  <xs:attribute name="STATUS" type="xs:string" />
                </xs:element>
             </xs:complexType>
           </xs:element>
        </xs:complexType>
      </xs:element>
     </xs:schema>
```