Using the VIP XML Import Module to Import Data from an Excel Spreadsheet

Article ID: 12366
Revision Date: 6-Jul-06

Summary

The XML Import Module allows importing of XML data from other applications or data sources for use as real-time numeric registers within the VIP.

A typical application might be to import data from an Excel spreadsheet. This data might represent, for example, scheduled production load for an industrial process. This data would be created by another program, or manually by a corporate manager, and the production manager would be responsible for managing loads to this schedule.

ION Enterprise can monitor real time loads and alarm on limits in order to manage loads, but these limits are typically static, or manually adjusted in most applications. To manually adjust these limits every day, hour or demand interval would be too cumbersome to manage. This example uses the XML Import Module to parse data from an Excel spreadsheet into the VIP. Once available to the VIP, scheduled alarming and control frameworks can be developed to automate the facility's load management.

Keywords

VIP XML Import module
Excel

Required Excel Conifguration

The Excel file must be exported to an XML file. This is done through "File" → "Save As" → then choose *.XML as the file type.

Required XML Import Module Conguration

XIM*n* URL

The file path/name of the XML file. An example could be:

d:\schedule.xml

XIMn Namespace

The namespace for the data elements to be parsed by the XIM. Typically, this will be:

xmlns:ss='urn:schemas-microsoft-com:office:spreadsheet'

This can be found in the XML file's header, when viewed in a common text editor such as Notepad. Note that when Excel creates the XML file, the

namespace line uses double quotes as delimiters as opposed to single quotes as in the example above. The XIM does not allow double quotes, so change these to single quotes.

XIM*n* xPath Query *n*

The xPath query to the desired data in the XML file. Excel arranges cell data in the hierarchy:

Workbook
Worksheet
Table
Row
Cell

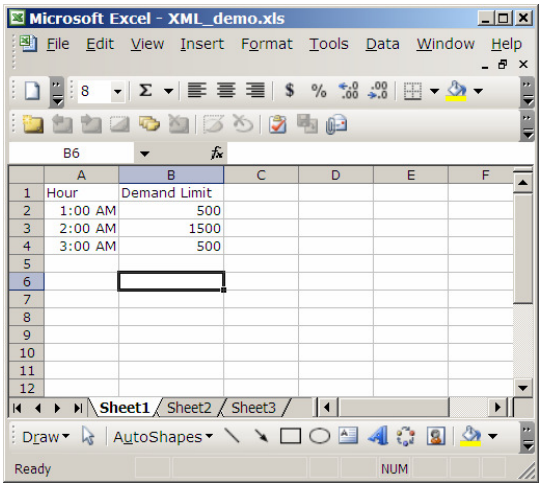
So the appropriate xPath would be of the form:

/ss:Workbook[*a*]/ss:Worksheet[*b*]/ss:Table[*c*]/ss:Row[*d*]/ss:Cell[*e*]/ss:Data

Where *a*, *b*, *c*, *d*, and *e* are the appropriate indices to the desired data.

EXAMPLE

For the following spreadsheet:



The corresponding XML file consists of:

```
<?xml version="1.0"?>
<?mso-application progid="Excel.Sheet"?>
<Workbook xmlns="urn:schemas-microsoft-com:office:spreadsheet"
 xmlns:o="urn:schemas-microsoft-com:office:office"
 xmlns:x="urn:schemas-microsoft-com:office:excel"
 xmlns:ss="urn:schemas-microsoft-com:office:spreadsheet"
 xmlns:html="http://www.w3.org/TR/REC-html40">
 <DocumentProperties xmlns="urn:schemas-microsoft-com:office:office">
  <Author>kevinb</Author>
  <LastAuthor>kevinb</LastAuthor>
  <Created>2006-07-06T16:02:39Z</Created>
  <LastSaved>2006-07-06T16:10:53Z</LastSaved>
  <Company>Power Measurement</Company>
  <Version>11.6568</Version>
 </DocumentProperties>
 <ExcelWorkbook xmlns="urn:schemas-microsoft-com:office:excel">
```

```
  <WindowHeight>9840</WindowHeight>
  <WindowWidth>14592</WindowWidth>
  <WindowTopX>384</WindowTopX>
  <WindowTopY>60</WindowTopY>
  <ProtectStructure>False</ProtectStructure>
  <ProtectWindows>False</ProtectWindows>
 </ExcelWorkbook>
 <Styles>
  <Style ss:ID="Default" ss:Name="Normal">
   <Alignment ss:Vertical="Bottom"/>
   <Borders/>
   <Font ss:FontName="Verdana" ss:Size="8"/>
   <Interior/>
   <NumberFormat/>
   <Protection/>
  </Style>
  <Style ss:ID="s21">
   <NumberFormat ss:Format="Medium Time"/>
  </Style>
 </Styles>
 <Worksheet ss:Name="Sheet1">
  <Table ss:ExpandedColumnCount="2" ss:ExpandedRowCount="4" x:FullColumns="1"
   x:FullRows="1" ss:DefaultColumnWidth="43.199999999999996"
   ss:DefaultRowHeight="10.2">
   <Column ss:Index="2" ss:AutoFitWidth="0" ss:Width="60.6"/>
   <Row>
    <Cell><Data ss:Type="String">Hour</Data></Cell>
    <Cell><Data ss:Type="String">Demand Limit</Data></Cell>
   </Row>
   <Row>
    <Cell ss:StyleID="s21"><Data ss:Type="DateTime">1899-12-
31T01:00:00.000</Data></Cell>
    <Cell><Data ss:Type="Number">500</Data></Cell>
   </Row>
   <Row>
    <Cell ss:StyleID="s21"><Data ss:Type="DateTime">1899-12-
31T02:00:00.000</Data></Cell>
    <Cell><Data ss:Type="Number">1500</Data></Cell>
   </Row>
   <Row>
    <Cell ss:StyleID="s21"><Data ss:Type="DateTime">1899-12-
31T03:00:00.000</Data></Cell>
    <Cell><Data ss:Type="Number">500</Data></Cell>
   </Row>
  </Table>
  <WorksheetOptions xmlns="urn:schemas-microsoft-com:office:excel">
   <Selected/>
   <Panes>
    <Pane>
     <Number>3</Number>
     <ActiveRow>5</ActiveRow>
     <ActiveCol>1</ActiveCol>
    </Pane>
   </Panes>
   <ProtectObjects>False</ProtectObjects>
   <ProtectScenarios>False</ProtectScenarios>
  </WorksheetOptions>
 </Worksheet>
 <Worksheet ss:Name="Sheet2">
  <Table ss:ExpandedColumnCount="0" ss:ExpandedRowCount="0" x:FullColumns="1"
   x:FullRows="1" ss:DefaultColumnWidth="43.199999999999996"
   ss:DefaultRowHeight="10.2"/>
  <WorksheetOptions xmlns="urn:schemas-microsoft-com:office:excel">
   <ProtectObjects>False</ProtectObjects>
   <ProtectScenarios>False</ProtectScenarios>
  </WorksheetOptions>
 </Worksheet>
 <Worksheet ss:Name="Sheet3">
  <Table ss:ExpandedColumnCount="0" ss:ExpandedRowCount="0" x:FullColumns="1"
   x:FullRows="1" ss:DefaultColumnWidth="43.199999999999996"
   ss:DefaultRowHeight="10.2"/>
  <WorksheetOptions xmlns="urn:schemas-microsoft-com:office:excel">
   <ProtectObjects>False</ProtectObjects>
   <ProtectScenarios>False</ProtectScenarios>
  </WorksheetOptions>
 </Worksheet>
</Workbook>
```

To access the Demand Limit data (shown in bold in the above file) using XIM #1, the following settings would be used:

XIM1

d:\XML_demo.xml

XIM1 Namespace

xmlns:ss='urn:schemas-microsoft-com:office:spreadsheet'

XIM1 xPath Query 1

/ss:Workbook[1]/ss:Worksheet[1]/ss:Table[1]/ss:Row[2]/ss:Cell[2]/ss:Data

XIM1 xPath Query 2

/ss:Workbook[1]/ss:Worksheet[1]/ss:Table[1]/ss:Row[4]/ss:Cell[2]/ss:Data

XIM1 xPath Query 2

/ss:Workbook[1]/ss:Worksheet[1]/ss:Table[1]/ss:Row[4]/ss:Cell[2]/ss:Data

More Information

- The XIM must receive a pulse on its "Read Now" input to import data from the XML file.
- If the XML file is open in another program, the XIM might not be able to load the file successfully. In this case, the outputs will go Not Available, and the event log will contain the entry:

  "File load failed. Either file not found or did not parse."

  This is particularly note when the XML file is open in Excel or Word. While the XIM will successfully load the data if the XML file is open in other editors such as Notepad, it is recommended that the XML file be left closed during import operations.
- The data in the spreadsheet to be read must be in a numeric format, as the XIM maps that data to ION Numeric registers. Consider the time data in the demo file shown above. In the spreadsheet, cell A2 contains the value "1:00 AM". In the XML file, the corresponding entry is:

  <Cell ss:StyleID="s21"><Data ss:Type="DateTime">**1899-12-31T01:00:00.000**</Data></Cell>

  Attempting to read this data into the XIM will result in a Not Available result. Note that formatting the cell to a "General Number" in Excel prior to saving the file as an XML file will allow the value to be read successfully, although the meaning of the number will still require decoding within the VIP.