Metasys® System Extended Architecture Secure Data Access DLL

Technical Bulletin

Code No. LIT-1201663
Software Release 5.0
Issued January 4, 2010
Supersedes October 6, 2008

Metasys® System Extended Architecture Secure Data Access DLL

Technical Bulletin

Document Introduction

This document describes how to use Web service technology to securely access the Metasys® system extended architecture. The document describes these services and how to install the Metasys System Secure Data Access Dynamic Link Library (DLL), use the Microsoft® Visual Basic® development system sample application, use the Microsoft Excel sample application, and use the ASP.NET sample application.

Note: The ASP.NET sample application can be installed only on a personal computer or server running the ADS/ADX software.

This document is intended for software developers who are familiar with DLL technology, and it assumes a working knowledge of Web services and any applications you intend to interface with the Metasys network.

Related Documentation

Table 1: Related Documentation

For Information On	See Document	LIT or Part Number
Installing the Application and Data Server (ADS), Extended Application and Data Server (ADX), and System Configuration Tool (SCT) Software	*ADS, ADX, and SCT Installation and Upgrade Instructions Wizard*[1]	*LIT-12011521*
Details on User Roles and Privileges	*Security Administrator System Technical Bulletin*	*LIT-1201528*
Metasys System Objects and Attributes	Metasys system *Help*	*LIT-1201793*[2]

1. Use the wizard to generate instructions specific to your system.
2. This LIT number represents a print friendly version of the Help.

Metasys System Secure Data Access DLL Overview

The Metasys System Secure Data Access DLL offers secure access to information in your Metasys network. The DLL (using the Web methods described in this document) allows you to build custom applications designed to quickly retrieve and update object attribute information in your Metasys network. Using custom interfaces created using the DLL and Web services, you can focus on specific tasks without learning how to use the entire Metasys system extended architecture User Interface (UI).

The Metasys System Secure Data Access DLL ensures that when you access Metasys system data with a custom application, you have the same high security as when you access the system data via the Metasys system UI. You must have appropriate network and application privileges required for a certain data request to be able to perform the request via a custom application. The Metasys System Secure Data Access DLL is required to use the BasicServices Web service.

Note: We recommend that if you write a custom application that uses a hard-coded user name and password (that is, it does not prompt a user to enter logon information through a user interface), then you should define a user specifically for that application. For example, for an energy monitoring application, you might create a user named EnergyApp. This allows you to adjust the privileges for this application independently of any human users or other applications that you write. It also improves the usefulness of the Audit Trail entries, because you can easily tell which application has interacted with the Metasys system.

Note: Details of Web service calls are proprietary. Users do not need to know the details of how Web service calls work to use the secure DLL.

Figure 1 shows an overview of how the three examples covered in this document query the Metasys system extended architecture using Web methods described in this document and the Metasys System Secure Data Access DLL.

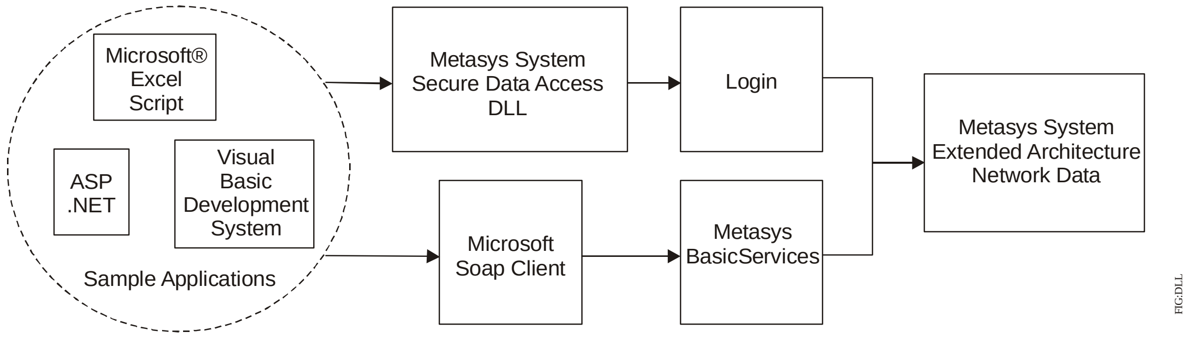

Microsoft® Excel Script
ASP .NET
Visual Basic Development System
Sample Applications
Metasys System Secure Data Access DLL
Login
Microsoft Soap Client
Metasys BasicServices
Metasys System Extended Architecture Network Data
FIG:DLL

Figure 1: Access to a Metasys System Using Web Services and the Metasys System Secure Data Access DLL

IMPORTANT: You must create the login user identification in the site director ADS/ADX **and** in each Network Automation Engine (NAE)/Network Integration Engine (NIE) read from (or written to) in the application that uses the secure DLL.

Create a security database with all the needed users and permissions. Back up this database and download it to any NAE/NIE accessed by the secure DLL.

Fully Qualified References

Every basic Web service requires a reference parameter; however, dealing with references can cause confusion. References sometimes play multiple roles, and it is not always clear which role the reference is being used for in each call.

When you are logged on to a Site Director, the Site Director looks at the first reference parameter of every call to determine which device to send the request to.

For some requests, the reference parameter is used only for routing to the proper device. See *ReadProperty Web Method*.

The comments on each method should help you sort it out.

Here are some general rules that always apply:

- If you are confused about the reference of an item, use the Metasys UI (if possible). Hold your cursor over an item in a navigation view to display a pop-up tool tip with the reference of that object. Also, the Advanced Focus view of most objects contains the Item Reference.
- A reference is a string that looks like `siteDirectorComputerName:deviceComputerName/item1.child1.grandchild1`

  The smallest valid reference contains just the siteDirectorComputerName and deviceComputerName pieces, and it is used to refer to a specific device on a site. Adding names to the right of the / allows you to refer to a specific object on a device.
- siteDirectorComputerName means the computer name of the Site Director.
- deviceComputerName means the computer name of the device to which you want to send the method. In the case of the Site Director, the deviceComputerName and the siteDirectorComputerName are the same thing.
- When you are logged on to a Site Director, the siteDirectorComputerName portion of your references must match the siteDirectorComputerName and the deviceComputerName portion of the reference must match one of the devices in the site.
- The siteDirectorComputerName portion of the object reference must match the site of which the object is a member, and the deviceComputerName portion of the object reference must match the deviceComputerName of the child device on to which you are logged. You may not obtain data from any other child device when logged on to a child device.

- There can be special references. Currently, there is only one. It is `siteDirectorComputerName:siteDirectorComputerName/$site`, where the **$site** specifies the Site object. The Site object is the one normally appearing at the top of the All Items Navigation View in the Metasys UI. The Site object is always on the Site Director (so the siteDirectorComputerName and deviceComputerName are identical).

Table 2: Valid References

Reference	Explanation
ads35:ads35	Refers to the device with the computer name ads35 on the site whose site director has the computer name ads35. **Note:** This refers to the Site Director.
ads35:ads35/$site	Refers to the Site object which is on the site director with the computer name ads35.
ads35:ads35/Programming	Refers to the object named Programming on the device with the computer name ads35.
ads35:nae3	Refers to the device with the computer name nae3 on the site whose site director has the computer name nae3.
ads35:nae3/Programming.AV1	Refers to the object named AV1 under an object named Programming on a device with the computer name nae3, which is a child of the site director with a computer name ads35.

Table 3: Invalid References

Reference	Explanation
ads35:nae23/$site	Specifies a Site object on a child device. A child device cannot have a Site object.
ads35/Programming	Lacks the siteDirectorComputerName:deviceComputerName format.
ads35	Lacks the siteDirectorComputerName:deviceComputerName portion of the reference.

Initialization

Before calling the BasicServices Web service, you must first initialize several Metasys System Secure Data Access classes. Whenever you create a new instance of the MSSOAP.SoapClient30 class, you also need to perform a security initialization. Before making any BasicServices Web method calls, create an instance of the MetasysSystemSecureDataAccess.MSSDAAPI class. Call its LoginUser method, passing it the hostname of the device to which you intend to log on and a valid user name and password pair. This method performs functionality similar to logging on to the Metasys UI. It attempts to log on to the device indicated in the first parameter of this method (typically the Site Director). After successfully logging on, you must set the HeaderHandler property of your SoapClient30 instance to be the value of the HeaderHandler property of your MSSDAAPI instance. For example:

```
myMSSDAAPI.LoginUser("MySite", "UserName", "Password", retStatus)
mySoapClient.HeaderHandler = myMSSDAAPI.HeaderHandler
```

This header handler adds Simple Object Access Protocol (SOAP) headers to each subsequent Web method call that proves to the Metasys system that you have logged on successfully.

In addition to the one-time initialization of the SoapClient30 class previously described, each time you call a particular Web method, you must first call the InitMethodAuthentication method of your MSSDAAPI instance to initialize additional security handling for that particular Web method call. For example,

```
myMSSDAAPI.InitMethodAuthentication("2003-06-28T13:26:13Z", _
"GetDeviceList", "MySite:MySite", retStatus)
```

The most significant parameter to the InitMethodAuthentication method is the first parameter that takes a string containing Universal Time Coordinated (UTC) time in International Organization for Standardization (ISO) 8601 format. This time must be the same (within 5 minutes) as the time of the Metasys system site with which you are communicating, or the Web service authentication fails. Call the GetCurrentTime Web method of the TimeService Web service to read the UTC time from the Metasys system site. The second parameter is the name of the Web method being called. The third parameter is the value passed to the Web method for its input parameter named Reference. Each of the Web methods takes a parameter named reference that the Metasys system uses to route the request to the appropriate device to service the call. The fourth parameter returns a status string.

The sample applications perform all this initialization, so you can copy that code to get started quickly. The Visual Basic and Excel sample applications assume that they are running on a computer that is not configured as a device on the Metasys system site, so they perform the GetCurrentTime method call mentioned previously; however, the ASP.NET sample application assumes that it is running on an ADS or ADX that is configured as a part of the Metasys system site, so it uses the local ADS or ADX computer’s representation of UTC time.

Web Services

Web services are collections of functions that allow data exchange among different software applications over networks. Web services are invoked using a standard protocol such as SOAP, an Extensible Markup Language (XML) based protocol. For example, the GetDeviceList Web method retrieves a list of all devices on the Metasys system extended architecture from the Site Director, without requiring Metasys system UI access.

The BasicServices Web service covered in this document is specific to the Metasys system extended architecture. In this document, the term attribute is synonymous with property, and text that identifies these properties is the exact text from the Metasys system extended architecture UI. The *Object Help* in the Metasys system *Help* is another reference for the text that represents attributes and commands. Typically, all Hypertext Transfer Protocol (HTTP) requests are sent from the custom application to the Site Director. In the method description tables, the deviceComputerName in the Endpoint Uniform Resource Locator (URL) usually is the hostname or Internet Protocol (IP) address of the Site Director.

To access the BasicServices Web service, you must install the Metasys System Secure Data Access DLL, because it handles user authentication for the Metasys system. For steps on installing the Metasys System Secure Data Access DLL. See *Detailed Procedures*.

GetCurrentTime Web Method

The GetCurrentTime Web method is used to read the time from a Metasys system device. This is important when a computer that needs to communicate with a Metasys system device does not have its time synchronized with the Metasys site. Table 4 details the GetCurrentTime Web method.

All the previous Web methods in this document are part of the BasicServices Web service. The GetCurrentTime Web method is part of the TimeService Web service and therefore has a different Endpoint URL.

Table 4: GetCurrentTime Web Method

Method Name	**GetCurrentTime**	
Endpoint URL	http://<deviceComputerName>/MetasysIII/WS/TimeManagement/TimeService.asmx	
SOAP Action	http://johnsoncontrols.com/MetasysIII/WebServices/GetCurrentTime	
Method Namespace URL	http://johnsoncontrols.com/MetasysIII/WebServices/	
Input Parameters	None	
Output Parameters	None	
Return Value	**Data Type**	**Description**
	XML	A time stamp containing a time_zone_enum element and a utc_date_time element
Method Signature	Function GetCurrentTime() As IXMLDOMNodeList	

Security Extension Check

There is no security checking done on the GetCurrentTime Web method.

Results

The time stamp XML contains a utc_date_time element. The value contained in this element is UTC time (in ISO 8601 format) as understood by the device.

For example:

```
<GetCurrentTimeResult>
<time_zone_enum>
<Metasys:enum set="576">53</Metasys:enum>
</time_zone_enum>
<utc_date_time>2003-06-22T03:00:23Z</utc_date_time>
</GetCurrentTimeResult>
```

In this example, **2003-06-22T03:00:12Z** is the string passed to the MSSDAAPI.InitMethodAuthentication method before calling one of the BasicServices Web methods. See the sample applications for more examples of parsing and generating updated UTC time strings.

Figure 2 shows an example of a GetCurrentTime Web method call.

```
Function GetCurrentSiteTime(siteDirectorComputerName As String) _
    As String
'VB 6.0 Example
'Requires References To:
' * Metasys System Secure Data Access (MSSDA) dll
'       from Johnson Controls
' * Microsoft XML
' * Microsoft Soap Type Library 3.0
    Dim TimeSoapClient As MSSOAPLib30.SoapClient30
    Dim wsdlURL As String
    wsdlURL = "http://" + siteDirectorComputerName + _
        "/MetasysIII/WS/TimeManagement/TimeService.asmx?wsdl"

    ' Create a new instance of SoapClient30 and
    ' then call MSSoapInit with the correct url of
    ' TimeService web service.
    Set TimeSoapClient = New SoapClient30
    TimeSoapClient.MSSoapInit wsdlURL

    'Call the GetCurrentTime web method to retrieve the device’s
    'current time. A successful call will return XML items in the
    'currentTimeXml.
    Dim currentTimeXml As IXMLDOMNodeList
    Set currentTimeXml = TimeSoapClient.GetCurrentTime

    'The second node in the list is the utc_date_time node.
    GetCurrentSiteTime = currentTimeXml.Item(1).Text
End Function
```

Figure 2: Visual Basic Development System Example of GetCurrentTime Web Method

GetDeviceList Web Method

The GetDeviceList Web method returns a list of all of the Metasys system supervisory controllers on the site and all devices known to the Site Director. The Endpoint URL for this Web method must be the Site Director. Table 5 details the GetDeviceList Web method.

This method returns data only when you are logged on to the Site Director. If you are logged on to a child device, this Web method always returns an empty list.

The reference parameter for this method is used strictly for routing to the proper device. The only reference that makes sense is the reference of the Site Director because the Site Director is the only device that can service this request. To make these even easier, you can just pass an empty string (a string of length zero). Passing an empty string for the reference is the recommended approach.

Table 5: GetDeviceList Web Method

Method Name	**GetDeviceList**		
Endpoint URL	http://<deviceComputerName>/MetasysIII/WS/BasicServices.asmx[1]		
SOAP Action	http://johnsoncontrols.com/MetasysIII/WebServices/GetDeviceList		
Method Namespace URL	http://johnsoncontrols.com/MetasysIII/WebServices		
Input Parameter	**Parameter**	**Data Type**	**Description**
	reference	Text	The parameter should be set to an empty string or the fully qualified reference of the site director.
Output Parameter	**Parameter**	**Data Type**	**Description**
	deviceList	Set of text values	A set of text values, one for each device known to the Site Director
Return Value		Int32	0 = Success, otherwise Error (In case of error, the deviceList contains one entry, which is the error message.)
Method Signature	Function GetDeviceList(ByVal siteDirectorComputerName As String, ByRef DeviceList() As String) As Long		

1. The Computer Name of the Site Director

Security Extension Check

The Metasys system accepts requests from any valid Metasys system user who calls this Web method.

Results

The returned list of devices is a set of text values separated by commas in the following format:

```
<Item Reference>,<Node Classification>,<Object Type>
```

For example:

```
MySite:Device1,Device,NAE
```

The Node Classification is not an object attribute. It is a generalization of the Object Type and gives a more general description of the node than Object Type. The possible returned values are object, device, server, site, integration, controller, point, folder, reference, navList, and extension. The returned values are useful for interacting with certain classifications of objects.

Figure 3 and Figure 4 show an example of a GetDeviceList Web method call.

```
Private Sub GetDeviceList(siteDirectorComputerName As String, _
    userName As String, password As String)
'VB 6.0 Example
'Requires References To:
' * Metasys System Secure Data Access (MSSDA) dll
'       from Johnson Controls
' * Microsoft XML
' * Microsoft Soap Type Library 3.0

' This subroutine must be called with the
' computer name of the site director, or IP Address of the
' site director; and a valid userName and password

    Dim SOAPClient As SoapClient30 'Used to access web services
    Dim JCISecurity As MSSDAAPI    'Used to login to MSEA device
    Dim siteTime As String
    Dim reference As String
    Dim retStatus As String
    Dim result As Integer
    Dim deviceList() As String
    Dim device As String
    Dim deviceFields() As String
    Dim index As Integer


    ' It is recommended that the reference be an empty string
    ' when calling GetDeviceList
    reference = ""

    ' Login to site director
    Set JCISecurity = New MSSDAAPI
    result = JCISecurity.LoginUser(siteDirectorComputerName, _
        userName, password, retStatus)

    If result <> 0 Then
        GoTo Finish
    End If

    ' Create and initialize the soap client
    Set SOAPClient = New SoapClient30
    SOAPClient.MSSoapInit "http://" & siteDirectorComputerName & _
        "/MetasysIII/WS/BasicServices.asmx?WSDL"

    Set SOAPClient.headerHandler = JCISecurity.headerHandler
```

Figure 3: Visual Basic Development System Example of GetDeviceList Web Method Call (Part 1 of 2)

```
 ' You need the current site director time for security reasons
    ' GetCurrentSiteTime is defined elsewhere in
    ' MSSDA Technical Bulletin
    siteTime = GetCurrentSiteTime(siteDirectorComputerName)

    ' Initialize JCI Security for calling GetDeviceList web service
    JCISecurity.InitMethodAuthentication siteTime, "GetDeviceList", _
        reference, retStatus

    ' finally, call the GetDeviceList web service
    result = SOAPClient.GetDeviceList(reference, deviceList)

    If result <> 0 Then
        'DeviceList(0) contains an error message in this case
        retStatus = deviceList(0)
        GoTo Finish
    End If

    For index = 0 To UBound(deviceList)
        device = deviceList(index)
        MsgBox "Raw Device String: " & device
        deviceFields = Split(device, ",")
        MsgBox deviceFields(0) & " is a " & deviceFields(1) & _
            " of type " & deviceFields(2)
    Next

Finish:
    If result <> 0 Then
        MsgBox "UnexpectedError: " & retStatus
    End If
End Sub
```

Figure 4: Visual Basic Development System Example of GetDeviceList Web Method Call (Part 2 of 2)

GetObjectList Web Method

The GetObjectList Web method returns a list of all the directly nested items, including extensions, of the object reference. Table 6 details the GetObjectList Web method. The reference parameter for this method is used to route the request to the proper device and then to locate a specific object on the device. An empty string is acceptable for this parameter. An empty string refers to the top level object on the device you are logged on to. If you are logged on to a site director, an empty string refers to the site object, and therefore this method returns the child objects of the site object. If you are logged directly on to a child device, the top level object is the device object of the device you are logged on to.

Table 6: GetObjectList Web Method

Method Name	**GetObjectList**		
Endpoint URL	http://<deviceComputerName>/MetasysIII/WS/BasicServices.asmx		
SOAP Action	http://johnsoncontrols.com/MetasysIII/WebServices/GetObjectList		
Method Namespace URL	http://johnsoncontrols.com/MetasysIII/WebServices/		
Input Parameter	**Parameter**	**Data Type**	**Description**
	reference	Text	A fully-qualified reference to the object whose nested objects are to be returned. An empty string is also acceptable.
Output Parameter	**Parameter**	**Data Type**	**Description**
	objectList	Set of text values	Set of text values, one for each object that is a direct child of the object reference
Return Value		Int32	0 = Success, otherwise Error (In case of error, the deviceList contains one entry, which is the error message.)
Method Signature	Function GetObjectList(ByVal ObjectReference As String, ByRef ObjectList() As String) As Long		

Security Extension Check

The Metasys system accepts requests from any valid Metasys system user who calls this Web method.

Results

The returned list of objects is a set of text values separated by commas in the same format as the text values returned by the GetDeviceList method:

```
<Item Reference>,<Node Classification>,<Object Type>
```

for example:

```
MySite:Device1/N2Trunk1/VAV3/Zone Temperature,Point,AI
```

The Node Classification is not an object attribute. It is a generalization of the Object Type and gives a more general description of the node than Object Type. The possible returned values are object, device, server, site, integration, controller, point, folder, reference, navList, and extension. This is useful for operating on certain classifications of objects.

Figure 5 and Figure 6 show an example of a GetObjectList Web method call.

```
Private Sub GetObjectList(siteDirectorComputerName As String, _
    userName As String, password As String)
'VB 6.0 Example
'Requires References To:
' * Metasys System Secure Data Access (MSSDA) dll
'       from Johnson Controls
' * Microsoft XML
' * Microsoft Soap Type Library 3.0

' This subroutine must be called with the
' computer name of the site director, or IP Address of the
' site director; and a valid userName and password
    Dim SOAPClient As MSSOAPLib30.SoapClient30
    Dim JCISecurity As MSSDAAPI
    Dim siteTime As String
    Dim reference As String
    Dim result As Long
    Dim retStatus As String
    Dim ObjectList() As String
    Dim object As String
    Dim objectFields() As String
    Dim index As Integer

    reference = siteDirectorComputerName & ":" & _
        siteDirectorComputerName

    Set JCISecurity = New MSSDAAPI
    result = JCISecurity.LoginUser(siteDirectorComputerName, _
        userName, password, retStatus)

    If result <> 0 Then
        GoTo Finish
    End If

    Set SOAPClient = New SoapClient30
    SOAPClient.MSSoapInit "http://" & siteDirectorComputerName & _
        "/MetasysIII/WS/BasicServices.asmx?WSDL"
    Set SOAPClient.headerHandler = JCISecurity.headerHandler

    ' GetCurrentSiteTime is defined elsewhere in
    ' MSSDA Technical Bulletin
    siteTime = GetCurrentSiteTime(siteDirectorComputerName)
    JCISecurity.InitMethodAuthentication siteTime, "GetObjectList", _
        reference, retStatus

    result = SOAPClient.GetObjectList(reference, ObjectList)

    If result <> 0 Then
        'if error, then objectList(0) contains an error message
        retStatus = ObjectList(0)
        GoTo Finish
    End If
```

Figure 5: Visual Basic Development System Example of GetObjectList Web Method Call (Part 1 of 2)

```
    Dim msgBoxResult As VbMsgBoxResult
    For index = 0 To UBound(ObjectList)
        object = ObjectList(index)
        MsgBox "Raw Object String: " & object
        objectFields = Split(object, ",")
        msgBoxResult = MsgBox(objectFields(0) & " is a " & _
            objectFields(1) & " of type " & objectFields(2), _
            vbOKCancel)
        If msgBoxResult = vbCancel Then
            Exit For
        End If
    Next

Finish:
    If result <> 0 Then
        MsgBox "Unexpected Error: " & retStatus
    End If
End Sub
```

Figure 6: Visual Basic Development System Example of GetObjectList Web Method Call (Part 2 of 2)

ReadProperty Web Method

The ReadProperty Web method reads a single attribute (property) from a single object. If you need to read more than one attribute or the same attribute from more than one object on a device, use ReadPropertyMultiple. See *ReadPropertyMultiple Web Method* for details. Table 7 details the ReadProperty Web method.

Table 7: ReadProperty Web Method

Method Name	**ReadProperty**		
Endpoint URL	http://<deviceComputerName>/MetasysIII/WS/BasicServices.asmx		
SOAP Action	http://johnsoncontrols.com/MetasysIII/WebServices/ReadProperty		
Method Namespace URI	http://johnsoncontrols.com/MetasysIII/WebServices/		
Input Parameters	**Parameter**	**Data Type**	**Description**
	reference	Text	A fully qualified Item Reference to the object whose attribute value is to be returned
	property	Text	Text in the format `<PropertyName>,<ArrayIndex>` or `<PropertyName>`. The ArrayIndex is only required for attributes that are arrays of the supported data types. <PropertyName> is the exact text from the Metasys system UI.
Output Parameters	**Parameter**	**Data Type**	**Description**
	stringValue	Text	Text representation of the attribute value formatted according to Table 8. For indexed attributes, the value is the element at the specified index. Otherwise, it contains any error messages generated while trying to read the attribute.
	rawValue	Real value	Numeric representation of the attribute value according to Table 9. For indexed attributes, the value is the element at the specified index, unless no index is specified, in which case the value contains the size of the value set.
	reliability	Text	The reliability, if any, of the requested attribute
	Priority	Text	The write priority, if any, of the requested attribute
Return Value		Int32	0 = Success, else Error (In case of error, the stringValue parameter contains the error message.)
Method Signature	Function ReadProperty(ByVal RefName As String, ByVal PropertyName As String, ByRef StringValue As String, ByRef RawValue As Double, ByRef Reliability As String, ByRef Priority As String) As Long		

Table 8: ReadProperty StringValue Output Data Type Description

Data Type	String Returned
True or False	True or False
Number	Numeric value
Real Value	Real value of the attribute
Text	String of text
One State/Type/Mode from a Set	Text string from the Metasys UI that represents the current value of the attribute
Date	Date in the format **mm/dd/yyyy**[1]
Time	Time in the format **hh:mm:ss tt** where **tt** is AM or PM
Other Data Types	Text string “Unsupported Data Type”

1. For information on international date formats, see *Appendix: International Date Formats*

Table 9: ReadProperty RawValue Output Data Type Description

Data Type	Value Returned
True or False	1 (True) or 0 (False)
Number	Number
Real Value	Number
Text	0
One State/Type/Mode from a Set	Numeric ID that represents the current value of the attribute
Date	OLE Automation Date where the whole number portion is Julian Date, and fraction is the percentage of the day’s seconds that have elapsed (typically 0 for date attributes)
Time	OLE Automation Date where the whole number portion is Julian Date, and fraction is the percentage of the day’s seconds that this time represents (that is, seconds since midnight/86400)
Other Data Types	1

Security Extension Check

If you are granted authorization on the View privilege for each object called, then the Metasys system accepts requests from a custom application that uses this Web method.

Results

Figure 7 and Figure 8 show an example of a ReadProperty Web method call.

```
Private Sub ReadProperty(siteDirectorComputerName As String, _
    userName As String, password As String)
'VB 6.0 Example
'Requires References To:
' * Metasys System Secure Data Access (MSSDA) dll
'       from Johnson Controls
' * Microsoft XML
' * Microsoft Soap Type Library 3.0

' This subroutine must be called with the
' computer name of the site director, or IP Address of the
' site director; and a valid userName and password
    Dim SOAPClient As MSSOAPLib30.SoapClient30
    Dim JCISecurity As MSSDAAPI
    Dim siteTime As String
    Dim reference As String
    Dim result As Long
    Dim retStatus As String
    Dim property As String
    Dim stringValue As String
    Dim rawValue As Single
    Dim reliability As String
    Dim priority As String

    ' Set reference to an AV. This assumes you
    ' have an object named AV1 in the Programming folder
    ' of your device
    reference = siteDirectorComputerName & ":" & _
        siteDirectorComputerName & "/Programming.AV1"

    Set JCISecurity = New MSSDAAPI
    result = JCISecurity.LoginUser(siteDirectorComputerName, _
        userName, password, retStatus)

    If result <> 0 Then
        GoTo Finish
    End If

    Set SOAPClient = New SoapClient30
    SOAPClient.MSSoapInit "http://" & siteDirectorComputerName & _
        "/MetasysIII/WS/BasicServices.asmx?WSDL"
    Set SOAPClient.headerHandler = JCISecurity.headerHandler

    ' GetCurrentSiteTime is defined elsewhere in
    ' MSSDA Technical Bulletin
    siteTime = GetCurrentSiteTime(siteDirectorComputerName)
    JCISecurity.InitMethodAuthentication siteTime, _
        "ReadProperty", reference, retStatus

    ' Read the Present Value Property, which is a numeric value
    property = "Present Value"
    result = SOAPClient.ReadProperty(reference, property, _
        stringValue, rawValue, reliability, priority)
```

Figure 7: Visual Basic Development System Example of ReadProperty Web Method Call (Part 1 of 2)

```
If result <> 0 Then
        'if error, then propertyList(0) contains an error message
        retStatus = stringValue
        GoTo Finish
    End If

    MsgBox reference & " Raw Value is " & CStr(rawValue)
    MsgBox reference & " String Value is " & stringValue
    MsgBox reference & " Reliability is " & reliability
    MsgBox reference & " Priority is " & priority

Finish:
    If result <> 0 Then
        MsgBox "Unexpected Error: " & retStatus
    End If
End Sub
```

Figure 8: Visual Basic Development System Example of ReadProperty Web Method Call (Part 2 of 2)

ReadPropertyMultiple Web Method

The ReadPropertyMultiple Web method reads one or more attributes (properties) from one or more objects within a single device. The same set of attributes is read from each object. You must have access to all the requested objects or the request is denied.

ReadPropertyMultiple exists to improve the performance by minimizing the number of messages sent over the network. Without this method, client applications would often need to call ReadProperty repeatedly to read all the data required. ReadPropertyMultiple allows the client to make one round trip to each device to read all the data it needs from that device. This improves the efficiency of the communication between the Metasys system device and the custom application. Table 10 details the ReadPropertyMultiple Web method.

Table 10: ReadPropertyMultiple Web Method

Method Name	**ReadPropertyMultiple**		
Endpoint URL	http://<deviceComputerName>/MetasysIII/WS/BasicServices.asmx		
SOAP Action	http://johnsoncontrols.com/MetasysIII/WebServices/ReadPropertyMultiple		
Method Namespace URL	http://johnsoncontrols.com/MetasysIII/WebServices/		
Input Parameter	**Parameter**	**Data Type**	**Description**
	Reference	Text	A fully qualified Item Reference to the device containing the objects to be read
	ObjectList	Set of text values	A set of fully qualified Item References to the objects containing the attributes to be read **Note:** All objects specified in ObjectList must be in the device indicated in the reference parameter.
	PropertyList	Set of text values	A set of text values using the exact text from the Metasys system UI containing the attributes to be read
Output Parameter	**Parameter**	**Data Type**	**Description**
	ValueList	Set of text values	A set of text values containing the values of the attributes read by ReadPropertyMultiple. See for a description of the individual values.
Return Value		Int32	0 = Success, else Error (In case of error, the ValueList contains one entry, which is the error message.)
Method Signature	Function ReadPropertyMultiple(ByVal DeviceReference As String, ByRef ObjectList() As String, ByRef PropertyList() As String, ByRef ValueList() As String) As Long		

Security Extension Check

If you are granted authorization on the View privilege for each object called, then the Metasys system accepts requests from a custom application that uses this Web method. If you are not granted authorization for every object, no value is returned.

Results

Figure 9 and Figure 10 show an example of a ReadPropertyMultiple Web method call.

```
Sub ReadPropertyMultiple(siteDirectorComputerName As String, _
    userName As String, password As String)
'VB 6.0 Example
'Requires References To:
' * Metasys System Secure Data Access (MSSDA) dll
'       from Johnson Controls
' * Microsoft XML
' * Microsoft Soap Type Library 3.0

' This subroutine must be called with the
' computer name of the site director, or IP Address of the
' site director; and a valid userName and password
    Dim SOAPClient As MSSOAPLib30.SoapClient30
    Dim JCISecurity As MSSDAAPI
    Dim siteTime As String
    Dim result As Long
    Dim retStatus As String
    Dim deviceReference As String
    Dim ObjectList(1) 'An Array of 2 elements
    Dim PropertyList(1) 'An Array of 2 elements
    Dim ValueList() As String 'An Array of strings
    Dim Value As Variant

    Set JCISecurity = New MSSDAAPI
    result = JCISecurity.LoginUser(siteDirectorComputerName, _
        userName, password, retStatus)

    If result <> 0 Then
        GoTo Finish
    End If

    Set SOAPClient = New SoapClient30
    SOAPClient.MSSoapInit "http://" & siteDirectorComputerName & _
        "/MetasysIII/WS/BasicServices.asmx?WSDL"
    Set SOAPClient.headerHandler = JCISecurity.headerHandler

    'Set up the parameters for ReadPropertyMultiple
    deviceReference = siteDirectorComputerName & ":" & _
        siteDirectorComputerName
    ObjectList(0) = deviceReference & "/Programming.AV1"
    ObjectList(1) = deviceReference & "/Programming.AV2"
    PropertyList(0) = "Present Value"
    PropertyList(1) = "Status"
```

Figure 9: Visual Basic Development System Example of ReadPropertyMultiple Web Method (Part 1 of 2)

```
' GetCurrentSiteTime is defined elsewhere in
    ' MSSDA Technical Bulletin
    siteTime = GetCurrentSiteTime(siteDirectorComputerName)
    JCISecurity.InitMethodAuthentication siteTime, _
        "ReadPropertyMultiple", deviceReference, retStatus

    result = SOAPClient.ReadPropertyMultiple(deviceReference, _
        ObjectList, PropertyList, ValueList)

    If result <> 0 Then
        'ValueList(0) should contain an error message in this case
        retStatus = ValueList(0)
        GoTo Finish
    End If

    '2 Objects * 2 Properties = 4 Values returned
    For Each Value In ValueList
        MsgBox Value
    Next

    'Another way to display returned values:
    Dim PropertyCount As Integer
    Dim iIndex As Integer, jIndex As Integer
    PropertyCount = UBound(PropertyList) + 1
    For iIndex = LBound(ObjectList) To UBound(ObjectList)
        For jIndex = LBound(PropertyList) To UBound(PropertyList)
            MsgBox "The " & PropertyList(jIndex) & " of " & _
                ObjectList(iIndex) & " is: " & _
                ValueList((iIndex * PropertyCount) + jIndex)
        Next
    Next

Finish:
    If result <> 0 Then
        MsgBox "Unexpected Error: " & retStatus
    End If
End Sub
```

Figure 10: Visual Basic Development System Example of ReadPropertyMultiple Web Method (Part 2 of 2)

WriteProperty Web Method

The WriteProperty Web method writes a single attribute of a single object. Table 11 details the WriteProperty Web method.

Table 11: WriteProperty Web Method

Method Name	**WriteProperty**		
Endpoint URL	http://<deviceComputerName>/MetasysIII/WS/BasicServices.asmx		
SOAP Action	http://johnsoncontrols.com/MetasysIII/WebServices/WriteProperty		
Method Namespace URL	http://johnsoncontrols.com/MetasysIII/WebServices/		
Input Parameter	**Parameter**	**Data Type**	**Description**
	reference	Text	A fully qualified Item Reference to the object containing the attribute to be written
	property	Text	Text in the format “<PropertyName>,<ArrayIndex>” or “<PropertyName>”. The ArrayIndex is only required for attributes that are arrays of the supported data types. The PropertyName is the exact text from the Metasys system UI.
	newValue	Text	Text representation of the new value to write to the attribute. See Table 11.
	priority	Text	For prioritized attributes, the specific occurrence of the priority array to update for that attribute. Text is the exact text from the Metasys system UI.
Output Parameter	**Parameter**	**Data Type**	**Description**
	status	Text	If the property value was successfully updated, status reads OK. Otherwise, it contains any error messages generated while trying to update the attribute.
Return Value		Int32	0 = Success, else Error (In case of error, the status parameter contains the error message.)
Method Signature	Function WriteProperty (ByVal RefName As String, ByVal PropertyName As String,ByVal NewValue As String, ByVal Priority As String, ByRef Status As String) As Long		

Table 12: WriteProperty newValue Input Data Type Description

Data Type	**String Returned**
True or False	If the value is 0 (False) or the value is 1 (True)
One state/type/mode from a set	If the value is an integer, then the value is interpreted as the set ID.
Date	If the value is in one of the standard formats supported by the .NET DateTime Parse() method, the value is interpreted as such. Refer to Microsoft .NET literature for more information on formats. The Date format is an example of one of the supported formats. If the value is a real value, it is interpreted as an OLE Automation date/time with the Julian date in the whole number portion and the time of day contained in the fractional portion as the percentage of the day’s seconds that have elapsed.
Time	If the value is in one of the standard formats supported by the .NET DateTime Parse() method, the value is interpreted as such. The Time format is an example of one of the supported formats. If the value is a real value, it is interpreted as an OLE Automation date/time with the Julian date in the whole number portion and the time of day contained in the fractional portion as the percentage of the day’s seconds that have elapsed.
Other Data Types	Value is assumed to correlate directly with the underlying data type for the attribute being updated.

Results

Figure 11 and Figure 12 show an example of a WriteProperty Web method call .

```
Sub WriteProperty(siteDirectorComputerName As String, _
    userName As String, password As String)
'VB 6.0 Example
'Requires References To:
' * Metasys System Secure Data Access (MSSDA) dll
'       from Johnson Controls
' * Microsoft XML
' * Microsoft Soap Type Library 3.0

' This subroutine must be called with the
' computer name of the site director, or IP Address of the
' site director; and a valid userName and password
    Dim SOAPClient As MSSOAPLib30.SoapClient30
    Dim JCISecurity As MSSDAAPI
    Dim siteTime As String
    Dim reference As String
    Dim result As Long
    Dim retStatus As String
    Dim property As String
    Dim newValue As String
    Dim priority As String
    Dim status As String

    reference = siteDirectorComputerName & ":" & _
        siteDirectorComputerName & "/Programming.AV1"
    property = "Description"
    newValue = "Average Temperature"
    Set JCISecurity = New MSSDAAPI
    result = JCISecurity.LoginUser(siteDirectorComputerName, _
        userName, password, retStatus)

    If result <> 0 Then
        GoTo Finish
    End If


Set SOAPClient = New SoapClient30
    SOAPClient.MSSoapInit "http://" & siteDirectorComputerName & _
        "/MetasysIII/WS/BasicServices.asmx?WSDL"
    Set SOAPClient.headerHandler = JCISecurity.headerHandler
```

Figure 11: Visual Basic Development System Example of WriteProperty Web Method Call (Part 1 of 2)

```
     ' GetCurrentSiteTime is defined elsewhere in
    ' MSSDA Technical Bulletin
    siteTime = GetCurrentSiteTime(siteDirectorComputerName)
    JCISecurity.InitMethodAuthentication siteTime, _
        "WriteProperty", reference, retStatus

result = SOAPClient.WriteProperty(reference, property, _
        newValue, priority, status)
    If result <> 0 Then
        'status contains an error message in this case
        retStatus = status
        GoTo Finish
    End If

    MsgBox reference & ": Property " & property & _
        " successfully updated to " & newValue
Finish:
    If result <> 0 Then
        MsgBox "Unexpected Error: " & retStatus
    End If
End Sub
```

Figure 12: Visual Basic Development System Example of WriteProperty Web Method Call (Part 2 of 2)

Security Extension Check

If you are granted authorization on the Modify or Configure Items privileges, then the Metasys system accepts requests from a custom application that uses this Web method.

SendCommand Web Method

The SendCommand Web method sends a single command to a single object. Table 13 details the SendCommand Web method.

Table 13: SendCommand Web Method

Method Name	**SendCommand**		
Endpoint URL	http://<deviceComputerName>/MetasysIII/WS/BasicServices.asmx		
SOAP Action	http://johnsoncontrols.com/Metasys/WebServices/SendCommand		
Method Namespace URL	http://johnsoncontrols.com/Metasys/WebServices/		
Input Parameter	**Parameter**	**Data Type**	**Description**
	reference	Text	A fully qualified reference to the object containing the attribute to be written
	command	Text	The name of the command to send using the exact text from the Metasys system UI
	parameters	Set of text values	Set of text values that contains representations of the values of any parameters included with the command Parameter names are not specified. The rules governing parameter value presentation are identical to those applied when presenting the newValue parameter for the WriteProperty Web method.
	priority	Number	For prioritized attributes, the specific occurrence of the priority array to update for that attribute
Output Parameter	**Parameter**	**Data Type**	**Description**
	status	Text	If the command was successfully executed, status reads OK. Otherwise, it contains any error messages generated while trying to command the object.
	returnParameters	Set of text values	Set of text values that represents any return parameters
Return Value		Int32	0 = Success, else Error (In case of error, the status parameter contains the error message.)
Method Signature	Function SendCommand(ByVal RefName As String, ByVal CommandName As String, ParameterList() As String, ByVal Priority As String, ByRef Status As String, ByRef ReturnParams() As String) As Long		

Results

Figure 13 and Figure 14 show an example of a SendCommand Web method call.

```
Sub SendCommand(siteDirectorComputerName As String, _
    userName As String, password As String)
'VB 6.0 Example
'Requires References To:
' * Metasys System Secure Data Access (MSSDA) dll
'       from Johnson Controls
' * Microsoft XML
' * Microsoft Soap Type Library 3.0

' This subroutine must be called with the
' computer name of the site director, or IP Address of the
' site director; and a valid userName and password
    Dim SOAPClient As MSSOAPLib30.SoapClient30
    Dim JCISecurity As MSSDAAPI
    Dim siteTime As String
    Dim reference As String
    Dim result As Long
    Dim retStatus As String
    Dim command As String
    Dim parameters() As String ' Zero length array
    Dim priority As String
    Dim status As String
    Dim returnParams() As String

    reference = siteDirectorComputerName & ":" & _
        siteDirectorComputerName
    command = "Archive"
    Set JCISecurity = New MSSDAAPI
    result = JCISecurity.LoginUser(siteDirectorComputerName, _
        userName, password, retStatus)

    If result <> 0 Then
        GoTo Finish
    End If

    Set SOAPClient = New SoapClient30
    SOAPClient.MSSoapInit "http://" & siteDirectorComputerName & _
        "/MetasysIII/WS/BasicServices.asmx?WSDL"
    Set SOAPClient.headerHandler = JCISecurity.headerHandler

    ' GetCurrentSiteTime is defined elsewhere in
    ' MSSDA Technical Bulletin
    siteTime = GetCurrentSiteTime(siteDirectorComputerName)
    JCISecurity.InitMethodAuthentication siteTime, _
        "SendCommand", reference, retStatus

    result = SOAPClient.SendCommand(reference, command, _
        parameters, priority, status, returnParams)

    If result <> 0 Then
        'status contains an error message in this case
        retStatus = status
        GoTo Finish
    End If
```

Figure 13: Visual Basic Development System Example of SendCommand Web Method Call (Part 1 of 2)

```
    MsgBox reference & ": Command " & command & _
        " sent successfully."

Finish:
    If result <> 0 Then
        MsgBox "Unexpected Error: " & retStatus
    End If
End Sub
```

Figure 14: Visual Basic Development System Example of SendCommand Web Method Call (Part 2 of 2)

Security Extension Check

The SendCommand Web method maps to an action in a one-to-one relationship (one command results in one action). Check the Security Administrator System to see the privilege to which a particular action (command) belongs. For example, Release Operator Override belongs to the Intervene privilege. If you are granted authorization on the relevant privilege, then the Metasys system accepts requests from a custom application that uses this Web method. Refer to the *Security Administrator System Technical Bulletin (LIT-1201528)* for details on user roles and privileges.

Basic Data Types

To simplify the Metasys Basic Web Services, they support only a subset of the data types that are used within a Metasys system. Table 14 shows the supported data types.

Table 14: Supported Data Types

Data Type	String Used in GetPropertyList or GetCommandList
Boolean	boolean
Byte	unsignedByte
Unsigned Integer	unsignedShort
Unsigned Long Integer	unsignedLong
Signed Integer	short
Signed Long Integer	long
Float	float
Double Precision Float	double
String	string
Enum	The name of the enum followed by the keyword enum. (specifically, Attribute enum, Email Authentication Type enum)
Date	date
Time	time
Array	One of the strings above follow by the word array (specifically, boolean array, Attribute enum array, unsignedByte array, …)

Individual elements of Arrays of the supported datatypes can also be read and written by passing the ArrayIndex as described in the method definitions. If no ArrayIndex is passed on a read operation and the type is array, then the size of the array is returned. ArrayIndex is required for write operations.

Sample Applications

The System Configuration Tool (SCT) CD contains three sample applications, along with the Metasys System Secure Data Access DLL. The Metasys System Secure Data Access DLL is required for the sample applications to run. To use custom applications to retrieve data for your system, design and build applications similar to these sample applications.

Sample application users need to log on using a valid Metasys Network user name and password. The Visual Basic development system and Excel examples require the hostname (or IP address) of the Site Director. The Metasys system only processes a function request from a Web method if the user is authorized to perform the function.

Requirements for Using the Sample Applications

The following are the requirements for using the sample applications:

- Microsoft Visual Basic Version 6.0 development system (for building the Visual Basic Development System sample application only)
- Microsoft Excel 97 or later (for using the Microsoft Excel sample application only)
- Microsoft Internet Explorer Version 6.0 or later (for using the ASP.NET sample application only)
- .NET Framework Version 1.1 (on the ADS/ADX where the ASP.NET sample application gets installed)
- ADS or ADX (for using the ASP.NET sample application only)
- Microsoft Visual Basic .NET 2002 (for the ASP.NET sample application only)
- Microsoft SOAP Toolkit Version 3.0 (installed with the DLL). The sample applications require the MSSOAP.SOAPClient30 object contained in the kit to create SOAP messages (SOAP is used to make Web method calls).

Visual Basic Development System Sample Application

The Visual Basic development system example application reads data from various objects, averages the data, and writes the result to another object.

Microsoft Excel Sample Application

The Excel sample application performs calls to all six of the published Web methods available in the BasicServices Web service.

ASP.NET Sample Application

The Application Service Provider (ASP.NET) sample application is written in VisualBasic.NET. It provides a set of Web pages that allows you to navigate to an object and read/update its data. The application uses all six of the published methods in the BasicServices Web service.

Detailed Procedures

Installing the Metasys System Secure Data Access DLL

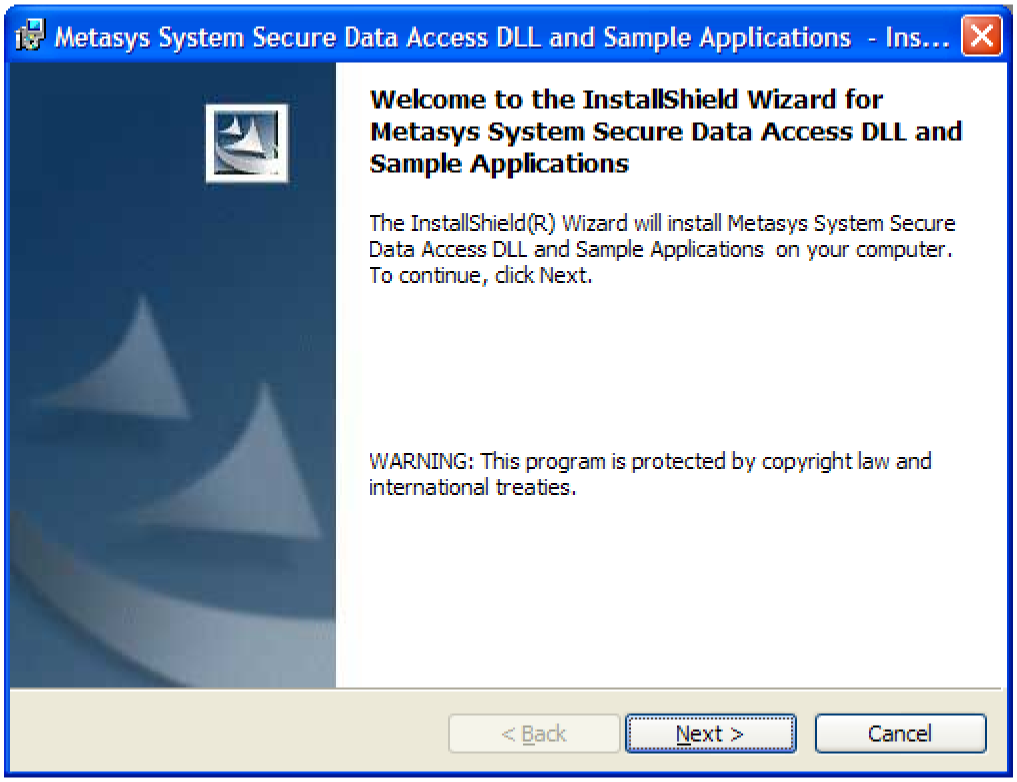
The ASP.NET sample application must be installed on a Metasys ADS or ADX configured as Site Director.

The Metasys System Secure Data Access DLL must be installed on an ADS or ADX.

Note: Do not install the Metasys System Secure Data Access DLL on a stand-alone SCT.

To install the Metasys System Secure Data Access DLL:

1. Insert the ADS, ADX, and SCT CD.
2. Navigate to the CD and browse to Setup.exe in the Metasys Sample Applications folder.
3. Double-click on Setup.exe. The Metasys System Secure Data Access DLL and Sample Applications Welcome screen appears (Figure 15).


Metasys System Secure Data Access DLL and Sample Applications - Ins...
< Back
Next >
Cancel

Welcome to the InstallShield Wizard for Metasys System Secure Data Access DLL and Sample Applications

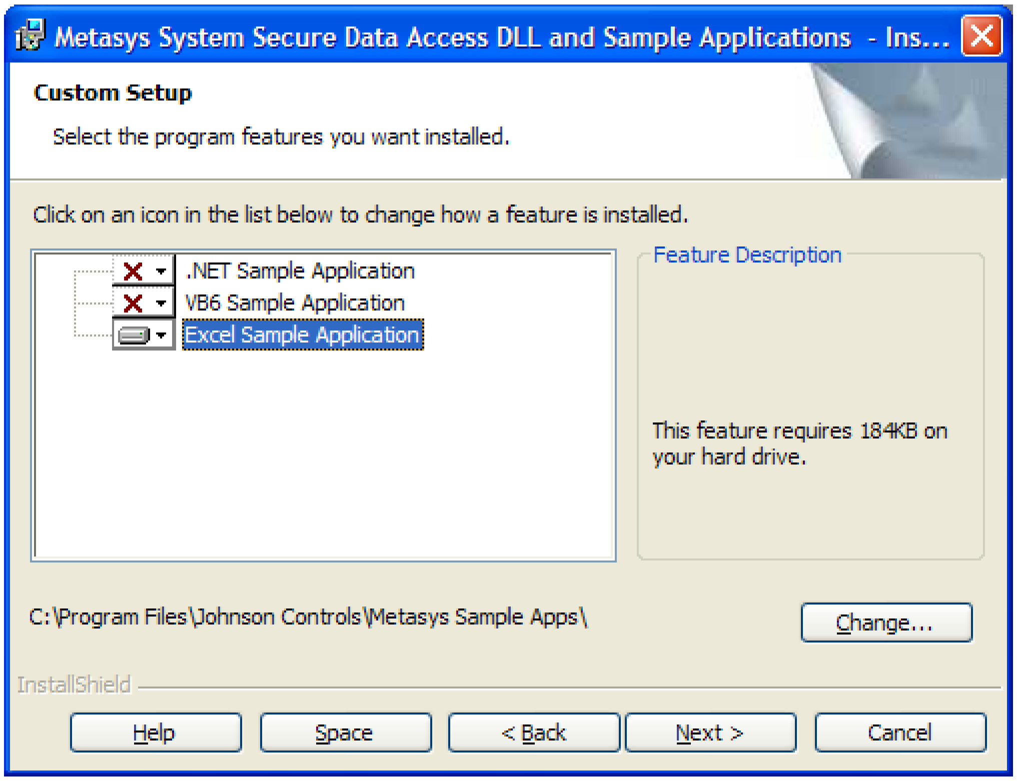
The InstallShield(R) Wizard will install Metasys System Secure Data Access DLL and Sample Applications on your computer. To continue, click Next.

WARNING: This program is protected by copyright law and international treaties.

Figure 15: Welcome Screen

4. Click Next and the License Agreement screen appears. Read the license agreement and click the radio button for **I accept the terms in this license agreement.**
5. Click Next. The Customer Information screen appears. Type in your information.

6. Click Next. The Custom Setup screen appears (Figure 16).


Metasys System Secure Data Access DLL and Sample Applications - Ins...

Custom Setup

Select the program features you want installed.

Click on an icon in the list below to change how a feature is installed.

- .NET Sample Application
- VB6 Sample Application
- Excel Sample Application

Feature Description

This feature requires 184KB on your hard drive.

C:\Program Files\Johnson Controls\Metasys Sample Apps\

Figure 16: Custom Setup Screen

7. Click on the icon next to the sample applications to select how you install them and click Next. The Ready to Install the Program screen appears.

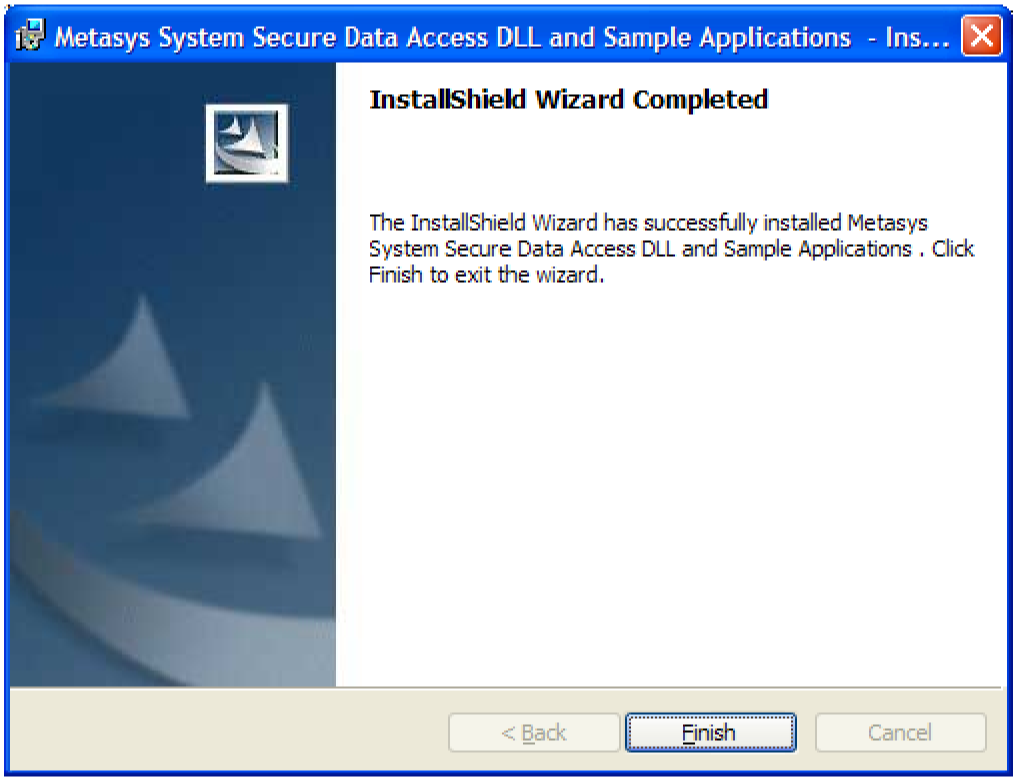
   Note: The ASP.NET sample application must be installed on a Metasys ADS or ADX configured as the Site Director.

8. Click Install and the Metasys System Secure Data Access DLL, sample applications, and Microsoft SOAP Toolkit Version 3.0 install on your computer. When the installation is complete, the InstallShield® Wizard Completed screen appears (Figure 17).

   Note: When you install the Metasys System Secure Data Access DLL, the Microsoft XML Core Services (MSXML) Version 4.0 Service Pack 2 English-specific module is installed and overwrites msxml4r.dll if it was originally installed with a service pack in a different language. To restore the local version of msxml4r.dll, update to your localized version of MSXML Version 4.0 Service Pack 2 (available from Microsoft Corporation) after installing Metasys System Secure Data Access.

   Note: To prevent the ADS from timing out, change the default Timeout property in the SOAP Toolkit to a larger number.

The default installation destination for the Metasys System Secure Data Access DLL and sample applications is C:\Program Files\Johnson Controls\Metasys Sample Apps. The sample applications install under subfolders.


Metasys System Secure Data Access DLL and Sample Applications - Ins...
< Back
Finish
Cancel

InstallShield Wizard Completed

The InstallShield Wizard has successfully installed Metasys System Secure Data Access DLL and Sample Applications . Click Finish to exit the wizard.

Figure 17: InstallShield Wizard Completed Screen

9. Click Finish.
10. Restart your computer.

Using the Visual Basic Development System Sample Application

To use a sample application, the Metasys System Secure Data Access DLL also must be installed on the same computer. See *Installing the Metasys System Secure Data Access DLL*.

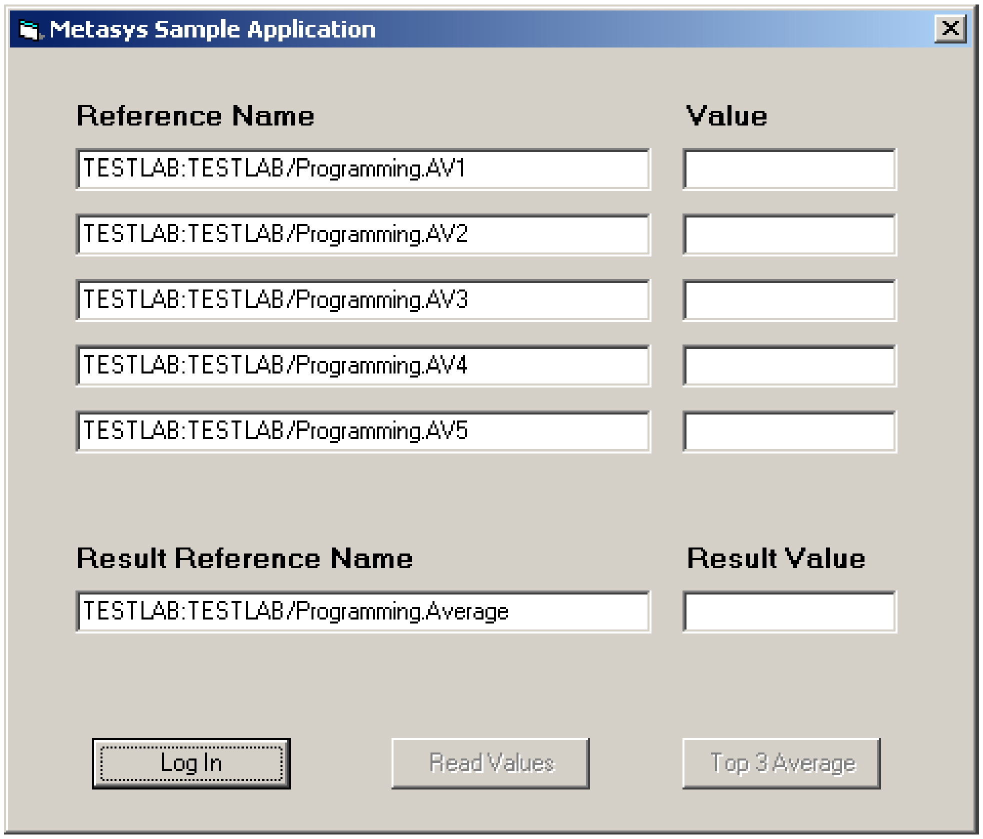
To use the Visual Basic development system sample application:

1. Browse to the location where you installed the Visual Basic development system sample application. The default location is C:\Program Files\Johnson Controls\Metasys Sample Apps.
2. Open the VB App folder.
3. Double-click on MetasysSampleApp.vbp. The sample application loads in Visual Basic Version 6.0.
4. On the Project menu, select the References… menu item and make sure that the Metasys System Secure Data Access check box is checked.
5. On the File menu, select Save.

6. On the File menu, select Make.
7. In Windows® Explorer, browse to and double-click MetasysSampleApp.exe in the VB App folder. The Visual Basic development system Main screen appears.
8. Click the Log In button. The Log In screen appears.

   Note: The login hostname/IP address of the Site Director is not case sensitive, but all other references are case sensitive.

9. Enter your Metasys system User Name (Login) and Password. Click OK. The Visual Basic development system main screen appears (Figure 18).



Reference Name	Value
TESTLAB:TESTLAB/Programming.AV1	
TESTLAB:TESTLAB/Programming.AV2	
TESTLAB:TESTLAB/Programming.AV3	
TESTLAB:TESTLAB/Programming.AV4	
TESTLAB:TESTLAB/Programming.AV5	

Result Reference Name	Result Value
TESTLAB:TESTLAB/Programming.Average	

Figure 18: Visual Basic Development System Main Screen

10. In the main screen (Figure 18), type in the Reference Names of the object attributes to be read. If you have used the Visual Basic development system sample application before, the attributes read last time are still entered.
11. Click the Read Values button to read the current values of the reference object attributes.
12. Type in the Result Reference Name of the attribute to which the Result Value is written.

13. Click the Top 3 Average button to do the following:
    a. Determine the top three values of the reference attributes.
    b. Calculate the average of the top three values.
    c. Display the average of the top three values in the Result Value text box.
    d. Write the value in Result Value to the Result Reference Name attribute.

Using the Excel Sample Application

To use a sample application, the Metasys System Secure Data Access DLL also must be installed on the same computer. See *Installing the Metasys System Secure Data Access DLL*.

To use the Excel sample application:

1. Browse to the location where you installed the Excel sample application. The default is C:\Program Files\Johnson Controls\Metasys Sample Apps.
2. Open the Excel App folder.
3. Double-click MetasysSampleApp.xls to open the Excel spreadsheet.
4. Click the Login tab at the bottom of the spreadsheet (Figure 19).

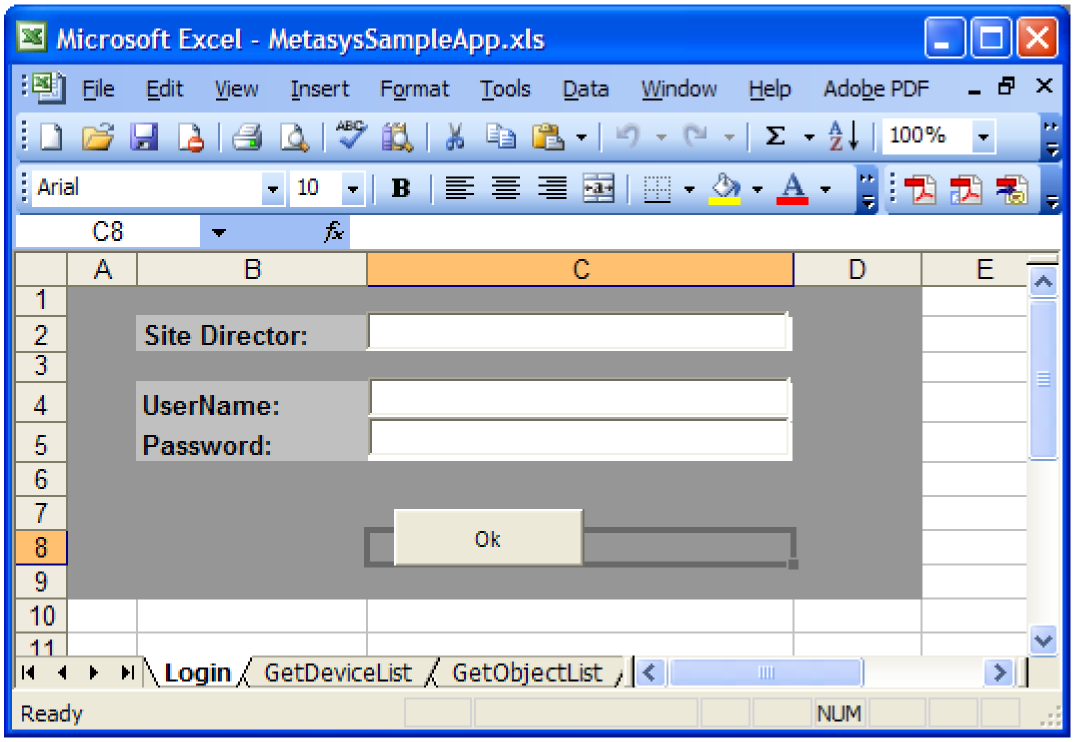

Microsoft Excel - MetasysSampleApp.xls
File Edit View Insert Format Tools Data Window Help Adobe PDF
100%
Arial 10
C8
Site Director:
UserName:
Password:
Ok
Login GetDeviceList GetObjectList
Ready NUM

Figure 19: Excel Login Sheet

5. On the Login sheet, enter the hostname (or IP address) of the Site Director and your Metasys system User Name and Password. (Note that the login hostname/ IP address of the Site Director is not case sensitive, but all other references are case sensitive.) Click OK. If the login is successful, a Login Succeeded message box appears.

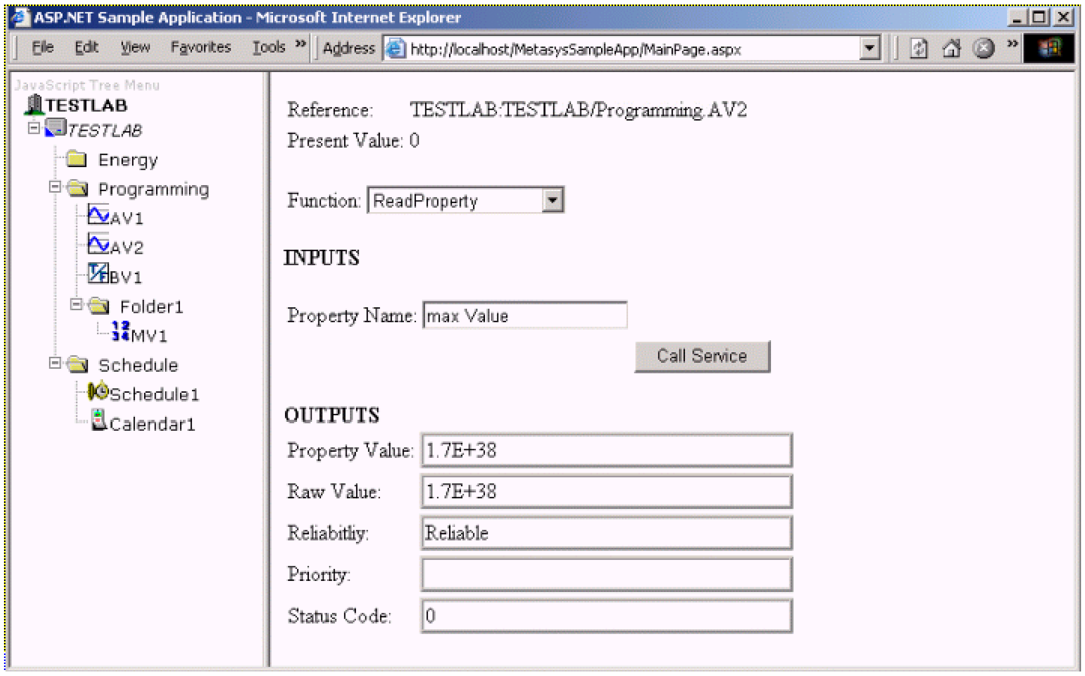
6. Click any of the tabs at the bottom of the spreadsheet to call the Web method of your choice. The available tabs are GetDeviceList, GetObjectList, ReadProperty, ReadPropertyMultiple, WriteProperty, and SendCommand.
7. Make the necessary changes on the sheet (for example, in the Reference cell) and click the button (for example, ReadProperty) to call the Web method.

Using the ASP.NET Sample Application

To use a sample application, the Metasys System Secure Data Access DLL also must be installed on the same computer. See the *Installing the Metasys System Secure Data Access DLL* section. The ASP.NET sample application must be installed on a Metasys ADS or ADX Site Director.

To use the ASP.NET sample application:

1. Enter the address of the ASP.NET page into the Web browser: http://<Site Director>/MetasysSampleApp/Login.aspx to start the ASP.NET sample application.
2. On the Log In screen, enter your Metasys system User Name and Password. Click Log In. (Note that the login hostname/IP address of the Site Director is not case sensitive, but all other references are case sensitive.) The main page Hypertext Markup Language (HTML) appears in the browser window (Figure 20).


ASP.NET Sample Application - Microsoft Internet Explorer
File Edit View Favorites Tools Address http://localhost/MetasysSampleApp/MainPage.aspx
JavaScript Tree Menu
TESTLAB
TESTLAB
Energy
Programming
AV1
AV2
BV1
Folder1
MV1
Schedule
Schedule1
Calendar1
Call Service

Reference: TESTLAB:TESTLAB/Programming.AV2
Present Value: 0

Function: ReadProperty

INPUTS

Property Name: max Value

OUTPUTS

Property Value: 1.7E+38
Raw Value: 1.7E+38
Reliabitliy: Reliable
Priority:
Status Code: 0

Figure 20: ASP.NET Main Page

This HTML is built by MainPage.aspx and contains two HTML frames. The frame on the left contains the HTML output from NavigationTree.aspx and the frame on the right contains HTML output from CommandCentral.aspx.

3. Select the desired object from the Navigation Tree on the left, expanding each folder as necessary. All folders and objects from the Metasys system extended architecture Navigation Tree appear in the application Navigation Tree.

   Note: The Reference and Present Value text boxes are filled in when an item in the Navigation Tree is selected. The OUTPUTS boxes are not editable and change depending on the function you call. GetDeviceList and GetObjectList provide the content for the Navigation Tree.

   If you select a device or folder, a separate page (generated by MultiRead.aspx) loads in the right frame that allows multiple attributes to be read from the objects directly nested below the selected device or folder with one ReadPropertyMultiple method call.

   When this sample application is run for the first time after being installed, the Navigation Tree is built. This requires multiple calls to GetObjectList which, for large sites, may take a significant amount of time to complete. Click the Rebuild Tree button (not shown) in the top right corner of the navigation frame to update the Navigation Tree.

4. Select the desired function (ReadProperty, ReadPropertyMultiple, WriteProperty, or SendCommand).
5. Enter appropriate values for the Inputs.
6. Click the Call Service button. See Table 15 for the possible responses.

Table 15: Call Service Button Responses

Function	Call Service Response
ReadProperty	The value of the Property Name text box is passed as an input parameter using the exact Metasys system UI. Outputs returned by the call are copied to Property Value, Raw Value, Reliability, Priority, and Status Code (represents the Return Value).
ReadPropertyMultiple	The value of the Property Name 1 and Property Name 2 text boxes under the INPUTS section are passed as input parameters. Outputs returned by the call are copied to Property Name 1, Property Value 1, Property Name 2, Property Value 2, and Status Code (represents the Return Value). If the Property Name 2 text box is left blank, only Property Value 1 is returned and Property Value 2 does not appear.
WriteProperty	The values of Property Name and Property Value boxes are passed as input parameters. Outputs returned by the call are copied to Status Code (represents the Return Value) and Status.
SendCommand	The values of Command, Parameter1, and Parameter2 are passed as input parameters. Outputs returned by the call are copied to Return Param 1, Return Param 2, and Status Code (represents the Return Value).

Troubleshooting

When you use the Microsoft Soap Client to make Web service requests, it waits 30 seconds for the Web server to respond. If the Web server does not respond in 30 seconds, the call times out, resulting in an error in your client. The server continues to process as normal, and it does not realize the client thinks an error occurred. Sometimes the first call to a Web service can take longer to process than subsequent calls, so sometimes the first call times out, while other ones do not.

You can change the default timeout for a request and give the server more time to process your call. Here is an example that sets the timeout to 1 minute:

```
Dim mySoapClient as MSSOAPLib30.SoapClient30
Set mySoapClient = New SoapClient30
' The following line will set the timeout to 1 minute.
' The timeout value is specified in milliseconds
mySoapClient.ConnectorProperty("Timeout") = 60000
```

Appendix: International Date Formats

Here is a table of Short Date Formats for various cultures and an example formatting for October 19, 2005.

Table 16: International Short Date Formats (Part 1 of 4)

ID	ISO Code	English Name	Short Date Pattern	Example
1025	ar	Arabic (Saudi Arabia)	dd/MM/yy	19/10/05
2049	ar	Arabic (Iraq)	dd/MM/yyyy	19/10/2005
3073	ar	Arabic (Egypt)	dd/MM/yyyy	19/10/2005
4097	ar	Arabic (Libya)	dd/MM/yyyy	19/10/2005
5121	ar	Arabic (Algeria)	dd-MM-yyyy	19-10-2005
6145	ar	Arabic (Morocco)	dd-MM-yyyy	19-10-2005
7169	ar	Arabic (Tunisia)	dd-MM-yyyy	19-10-2005
8193	ar	Arabic (Oman)	dd/MM/yyyy	19/10/2005
9217	ar	Arabic (Yemen)	dd/MM/yyyy	19/10/2005
10241	ar	Arabic (Syria)	dd/MM/yyyy	19/10/2005
11265	ar	Arabic (Jordan)	dd/MM/yyyy	19/10/2005
12289	ar	Arabic (Lebanon)	dd/MM/yyyy	19/10/2005
13313	ar	Arabic (Kuwait)	dd/MM/yyyy	19/10/2005
14337	ar	Arabic (U.A.E.)	dd/MM/yyyy	19/10/2005
15361	ar	Arabic (Bahrain)	dd/MM/yyyy	19/10/2005
16385	ar	Arabic (Qatar)	dd/MM/yyyy	19/10/2005
1026	bg	Bulgarian (Bulgaria)	dd.M.yyyy '?.'	19.10.2005 ?.
1027	ca	Catalan (Catalan)	dd/MM/yyyy	19/10/2005
1028	zh	Chinese (Taiwan)	yyyy/M/d	2005/10/19
2052	zh	Chinese (Peoples’ Republic of China)	yyyy-M-d	2005-10-19
3076	zh	Chinese (Hong Kong S.A.R.)	d/M/yyyy	19/10/2005
4100	zh	Chinese (Singapore)	d/M/yyyy	19/10/2005
5124	zh	Chinese (Macau S.A.R.)	d/M/yyyy	19/10/2005
1029	cs	Czech (Czech Republic)	d.M.yyyy	19.10.2005
1030	da	Danish (Denmark)	dd-MM-yyyy	19-10-2005
1031	de	German (Germany)	dd.MM.yyyy	19.10.2005
2055	de	German (Switzerland)	dd.MM.yyyy	19.10.2005
3079	de	German (Austria)	dd.MM.yyyy	19.10.2005
4103	de	German (Luxembourg)	dd.MM.yyyy	19.10.2005
5127	de	German (Liechtenstein)	dd.MM.yyyy	19.10.2005
1032	el	Greek (Greece)	d/M/yyyy	19/10/2005
1033	en	English (United States)	M/d/yyyy	10/19/2005
2057	en	English (United Kingdom)	dd/MM/yyyy	19/10/2005
3081	en	English (Australia)	d/MM/yyyy	19/10/2005
4105	en	English (Canada)	dd/MM/yyyy	19/10/2005
5129	en	English (New Zealand)	d/MM/yyyy	19/10/2005
6153	en	English (Ireland)	dd/MM/yyyy	19/10/2005
7177	en	English (South Africa)	yyyy/MM/dd	2005/10/19

Table 16: International Short Date Formats (Part 2 of 4)

ID	ISO Code	English Name	Short Date Pattern	Example
8201	en	English (Jamaica)	dd/MM/yyyy	19/10/2005
9225	en	English (Caribbean)	MM/dd/yyyy	10/19/2005
10249	en	English (Belize)	dd/MM/yyyy	19/10/2005
11273	en	English (Trinidad and Tobago)	dd/MM/yyyy	19/10/2005
12297	en	English (Zimbabwe)	M/d/yyyy	10/19/2005
13321	en	English (Republic of the Philippines)	M/d/yyyy	10/19/2005
2058	es	Spanish (Mexico)	dd/MM/yyyy	19/10/2005
3082	es	Spanish (Spain)	dd/MM/yyyy	19/10/2005
4106	es	Spanish (Guatemala)	dd/MM/yyyy	19/10/2005
5130	es	Spanish (Costa Rica)	dd/MM/yyyy	19/10/2005
6154	es	Spanish (Panama)	MM/dd/yyyy	10/19/2005
7178	es	Spanish (Dominican Republic)	dd/MM/yyyy	19/10/2005
8202	es	Spanish (Venezuela)	dd/MM/yyyy	19/10/2005
9226	es	Spanish (Colombia)	dd/MM/yyyy	19/10/2005
10250	es	Spanish (Peru)	dd/MM/yyyy	19/10/2005
11274	es	Spanish (Argentina)	dd/MM/yyyy	19/10/2005
12298	es	Spanish (Ecuador)	dd/MM/yyyy	19/10/2005
13322	es	Spanish (Chile)	dd-MM-yyyy	19-10-2005
14346	es	Spanish (Uruguay)	dd/MM/yyyy	19/10/2005
15370	es	Spanish (Paraguay)	dd/MM/yyyy	19/10/2005
16394	es	Spanish (Bolivia)	dd/MM/yyyy	19/10/2005
17418	es	Spanish (El Salvador)	dd/MM/yyyy	19/10/2005
18442	es	Spanish (Honduras)	dd/MM/yyyy	19/10/2005
19466	es	Spanish (Nicaragua)	dd/MM/yyyy	19/10/2005
20490	es	Spanish (Puerto Rico)	dd/MM/yyyy	19/10/2005
1035	fi	Finnish (Finland)	d.M.yyyy	19.10.2005
1036	fr	French (France)	dd/MM/yyyy	19/10/2005
2060	fr	French (Belgium)	d/MM/yyyy	19/10/2005
3084	fr	French (Canada)	yyyy-MM-dd	2005-10-19
4108	fr	French (Switzerland)	dd.MM.yyyy	19.10.2005
5132	fr	French (Luxembourg)	dd/MM/yyyy	19/10/2005
6156	fr	French (Principality of Monaco)	dd/MM/yyyy	19/10/2005
1037	he	Hebrew (Israel)	dd/MM/yyyy	19/10/2005
1038	hu	Hungarian (Hungary)	yyyy. MM. dd.	2005. 10. 19.
1039	is	Icelandic (Iceland)	d.M.yyyy	19.10.2005
1040	it	Italian (Italy)	dd/MM/yyyy	19/10/2005
2064	it	Italian (Switzerland)	dd.MM.yyyy	19.10.2005
1041	ja	Japanese (Japan)	yyyy/MM/dd	2005/10/19
1042	ko	Korean (Korea)	yyyy-MM-dd	2005-10-19
1043	nl	Dutch (Netherlands)	d-M-yyyy	19-10-2005
2067	nl	Dutch (Belgium)	d/MM/yyyy	19/10/2005

Table 16: International Short Date Formats (Part 3 of 4)

ID	ISO Code	English Name	Short Date Pattern	Example
1044	nb	Norwegian (Bokm†l) (Norway)	dd.MM.yyyy	19.10.2005
2068	nn	Norwegian (Nynorsk) (Norway)	dd.MM.yyyy	19.10.2005
1045	pl	Polish (Poland)	yyyy-MM-dd	2005-10-19
1046	pt	Portuguese (Brazil)	d/M/yyyy	19/10/2005
2070	pt	Portuguese (Portugal)	dd-MM-yyyy	19-10-2005
1048	ro	Romanian (Romania)	dd.MM.yyyy	19.10.2005
1049	ru	Russian (Russia)	dd.MM.yyyy	19.10.2005
1050	hr	Croatian (Croatia)	d.M.yyyy	19.10.2005
2074	sr	Serbian (Latin) (Serbia)	d.M.yyyy	19.10.2005
3098	sr	Serbian (Cyrillic) (Serbia)	d.M.yyyy	19.10.2005
1051	sk	Slovak (Slovakia)	d. M. yyyy	19. 10. 2005
1052	sq	Albanian (Albania)	yyyy-MM-dd	2005-10-19
1053	sv	Swedish (Sweden)	yyyy-MM-dd	2005-10-19
2077	sv	Swedish (Finland)	d.M.yyyy	19.10.2005
1054	th	Thai (Thailand)	d/M/yyyy	19/10/2005
1055	tr	Turkish (Turkey)	dd.MM.yyyy	19.10.2005
1056	ur	Urdu (Islamic Republic of Pakistan)	dd/MM/yyyy	19/10/2005
1057	id	Indonesian (Indonesia)	dd/MM/yyyy	19/10/2005
1058	uk	Ukrainian (Ukraine)	dd.MM.yyyy	19.10.2005
1059	be	Belarusian (Belarus)	dd.MM.yyyy	19.10.2005
1060	sl	Slovenian (Slovenia)	d.M.yyyy	19.10.2005
1061	et	Estonian (Estonia)	d.MM.yyyy	19.10.2005
1062	lv	Latvian (Latvia)	yyyy.MM.dd.	2005.10.19.
1063	lt	Lithuanian (Lithuania)	yyyy.MM.dd	2005.10.19
1065	fa	Farsi (Iran)	M/d/yyyy	10/19/2005
1066	vi	Vietnamese (Viet Nam)	dd/MM/yyyy	19/10/2005
1067	hy	Armenian (Armenia)	dd.MM.yyyy	19.10.2005
1068	az	Azeri (Latin) (Azerbaijan)	dd.MM.yyyy	19.10.2005
2092	az	Azeri (Cyrillic) (Azerbaijan)	dd.MM.yyyy	19.10.2005
1069	eu	Basque (Basque)	yyyy/MM/dd	2005/10/19
1071	mk	FYRO Macedonian (Former Yugoslav Republic of Macedonia)	dd.MM.yyyy	19.10.2005
1078	af	Afrikaans (South Africa)	yyyy/MM/dd	2005/10/19
1079	ka	Georgian (Georgia)	dd.MM.yyyy	19.10.2005
1080	fo	Faroese (Faroe Islands)	dd-MM-yyyy	19-10-2005
1081	hi	Hindi (India)	dd-MM-yyyy	19-10-2005
1086	ms	Malay (Malaysia)	dd/MM/yyyy	19/10/2005
2110	ms	Malay (Brunei Darussalam)	dd/MM/yyyy	19/10/2005
1087	kk	Kazakh (Kazakhstan)	dd.MM.yyyy	19.10.2005
1088	ky	Kyrgyz (Kyrgyzstan)	dd.MM.yy	19.10.05
1089	sw	Swahili (Kenya)	M/d/yyyy	10/19/2005
1091	uz	Uzbek (Latin) (Uzbekistan)	dd/MM yyyy	19/10 2005

Table 16: International Short Date Formats (Part 4 of 4)

ID	ISO Code	English Name	Short Date Pattern	Example
2115	uz	Uzbek (Cyrillic) (Uzbekistan)	dd.MM.yyyy	19.10.2005
1092	tt	Tatar (Russia)	dd.MM.yyyy	19.10.2005
1094	pa	Punjabi (India)	dd-MM-yy	19-10-05
1095	gu	Gujarati (India)	dd-MM-yy	19-10-05
1097	ta	Tamil (India)	dd-MM-yyyy	19-10-2005
1098	te	Telugu (India)	dd-MM-yy	19-10-05
1099	kn	Kannada (India)	dd-MM-yy	19-10-05
1102	mr	Marathi (India)	dd-MM-yyyy	19-10-2005
1103	sa	Sanskrit (India)	dd-MM-yyyy	19-10-2005
1104	mn	Mongolian (Mongolia)	yy.MM.dd	05.10.19
1110	gl	Galician (Galician)	dd/MM/yy	19/10/05
1111	kok	Konkani (India)	dd-MM-yyyy	19-10-2005
1114	syr	Syriac (Syria)	dd/MM/yyyy	19/10/2005
1125	div	Divehi (Maldives)	dd/MM/yy	19/10/05
127	iv	Invariant Language (Invariant Country)	MM/dd/yyyy	10/19/2005

Johnson Controls

Building Efficiency
507 E. Michigan Street, Milwaukee, WI 53202

Metasys® and Johnson Controls® are registered trademarks of Johnson Controls, Inc.
All other marks herein are the marks of their respective owners. © 2010 Johnson Controls, Inc.

Published in U.S.A. www.johnsoncontrols.com