Connecting to the WinMax Control using a Vendor ID and WCF Data Services

Hurco Companies, Inc.

Record of Changes

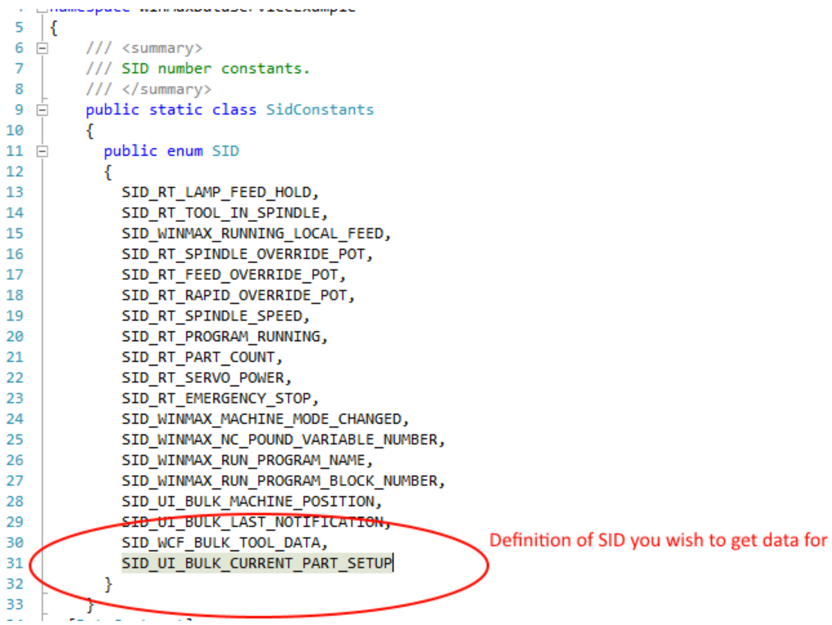
Rev.	Revision Description	Rev By	Date
1	Original Release	Ian Roberts, Paul Gray	2016.03.28

Introduction

This document refers to the software provided in the sample integration package. This connection is only available on Mill machines running v9 or higher control software and is presently not available on Lathes.

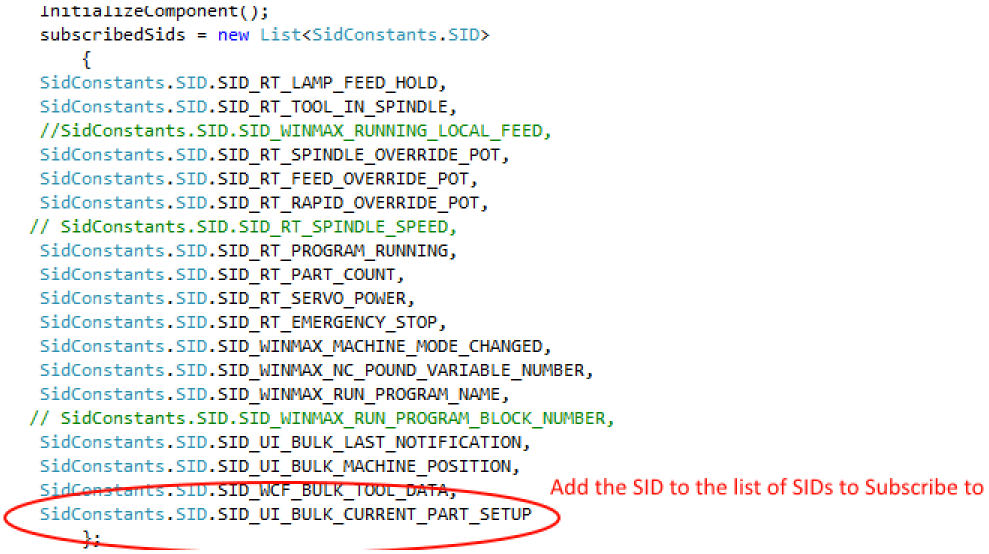
To connect to the Mill machines, the host CNC machine will need the Ultimonitor option along with the specific Vendor's Option. The Vendor will embed their Vendor ID and Vendor Password in their connection software as shown in the sample integration below. The Hurco control validates the connection based on this data and the presence of the Vendor Option.

1. **Add the SIDs for the data you require to the list in the file SidConstants.cs**
   SIDs are named Source IDs that your application subscribes to from the Hurco control. Your application will be pushed the associated data from the SIDs when the data changes.



```
namespace WinmaxDataServiceExample
{
    /// <summary>
    /// SID number constants.
    /// </summary>
    public static class SidConstants
    {
      public enum SID
      {
        SID_RT_LAMP_FEED_HOLD,
        SID_RT_TOOL_IN_SPINDLE,
        SID_WINMAX_RUNNING_LOCAL_FEED,
        SID_RT_SPINDLE_OVERRIDE_POT,
        SID_RT_FEED_OVERRIDE_POT,
        SID_RT_RAPID_OVERRIDE_POT,
        SID_RT_SPINDLE_SPEED,
        SID_RT_PROGRAM_RUNNING,
        SID_RT_PART_COUNT,
        SID_RT_SERVO_POWER,
        SID_RT_EMERGENCY_STOP,
        SID_WINMAX_MACHINE_MODE_CHANGED,
        SID_WINMAX_NC_POUND_VARIABLE_NUMBER,
        SID_WINMAX_RUN_PROGRAM_NAME,
        SID_WINMAX_RUN_PROGRAM_BLOCK_NUMBER,
        SID_UI_BULK_MACHINE_POSITION,
        SID_UI_BULK_LAST_NOTIFICATION,
        SID_WCF_BULK_TOOL_DATA,
        SID_UI_BULK_CURRENT_PART_SETUP
      }
    }
```

2. Add those SIDs to the list that the application will subscribe to

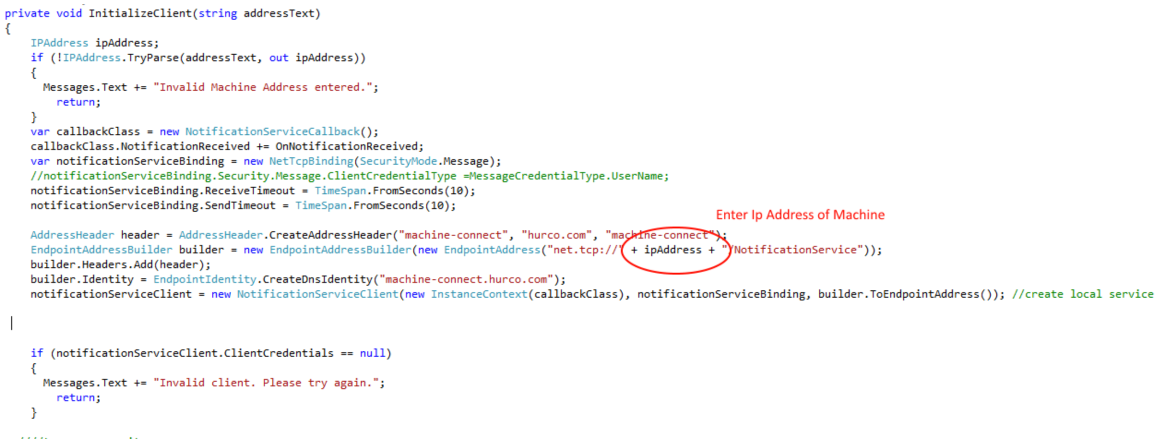


```
InitializeComponent();
subscribedSids = new List<SidConstants.SID>
    {
SidConstants.SID.SID_RT_LAMP_FEED_HOLD,
SidConstants.SID.SID_RT_TOOL_IN_SPINDLE,
//SidConstants.SID.SID_WINMAX_RUNNING_LOCAL_FEED,
SidConstants.SID.SID_RT_SPINDLE_OVERRIDE_POT,
SidConstants.SID.SID_RT_FEED_OVERRIDE_POT,
SidConstants.SID.SID_RT_RAPID_OVERRIDE_POT,
// SidConstants.SID.SID_RT_SPINDLE_SPEED,
SidConstants.SID.SID_RT_PROGRAM_RUNNING,
SidConstants.SID.SID_RT_PART_COUNT,
SidConstants.SID.SID_RT_SERVO_POWER,
SidConstants.SID.SID_RT_EMERGENCY_STOP,
SidConstants.SID.SID_WINMAX_MACHINE_MODE_CHANGED,
SidConstants.SID.SID_WINMAX_NC_POUND_VARIABLE_NUMBER,
SidConstants.SID.SID_WINMAX_RUN_PROGRAM_NAME,
// SidConstants.SID.SID_WINMAX_RUN_PROGRAM_BLOCK_NUMBER,
SidConstants.SID.SID_UI_BULK_LAST_NOTIFICATION,
SidConstants.SID.SID_UI_BULK_MACHINE_POSITION,
SidConstants.SID.SID_WCF_BULK_TOOL_DATA,
SidConstants.SID.SID_UI_BULK_CURRENT_PART_SETUP
    };
```

Add the SID to the list of SIDs to Subscribe to

3. Initialize the connections to WinMax.

Here the sample integration uses an IP address from a textbox.

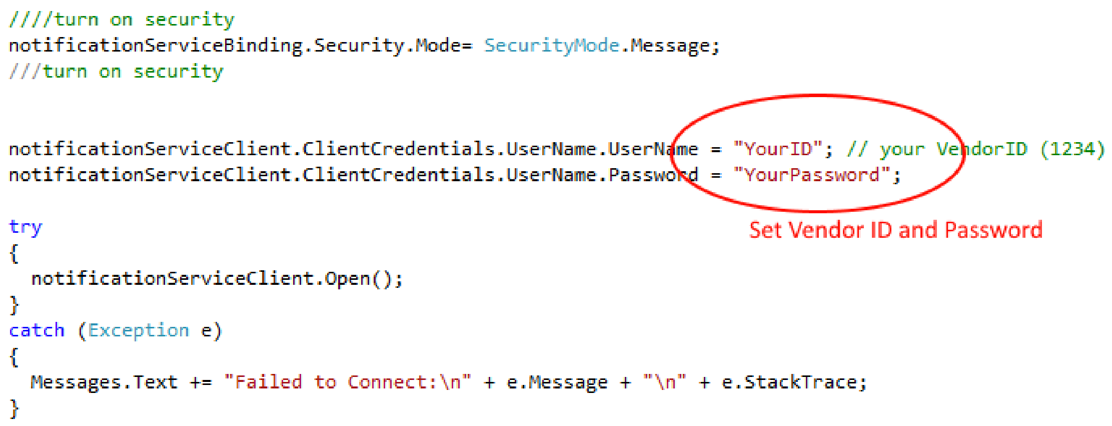


```
private void InitializeClient(string addressText)
{
    IPAddress ipAddress;
    if (!IPAddress.TryParse(addressText, out ipAddress))
    {
      Messages.Text += "Invalid Machine Address entered.";
        return;
    }
    var callbackClass = new NotificationServiceCallback();
    callbackClass.NotificationReceived += OnNotificationReceived;
    var notificationServiceBinding = new NetTcpBinding(SecurityMode.Message);
    //notificationServiceBinding.Security.Message.ClientCredentialType =MessageCredentialType.UserName;
    notificationServiceBinding.ReceiveTimeout = TimeSpan.FromSeconds(10);
    notificationServiceBinding.SendTimeout = TimeSpan.FromSeconds(10);

    AddressHeader header = AddressHeader.CreateAddressHeader("machine-connect", "hurco.com", "machine-connect");
    EndpointAddressBuilder builder = new EndpointAddressBuilder(new EndpointAddress("net.tcp://" + ipAddress + "/NotificationService"));
    builder.Headers.Add(header);
    builder.Identity = EndpointIdentity.CreateDnsIdentity("machine-connect.hurco.com");
    notificationServiceClient = new NotificationServiceClient(new InstanceContext(callbackClass), notificationServiceBinding, builder.ToEndpointAddress()); //create local service


    if (notificationServiceClient.ClientCredentials == null)
    {
      Messages.Text += "Invalid client. Please try again.";
        return;
    }
```

Enter Ip Address of Machine

4. **Set your Vendor ID and Password**

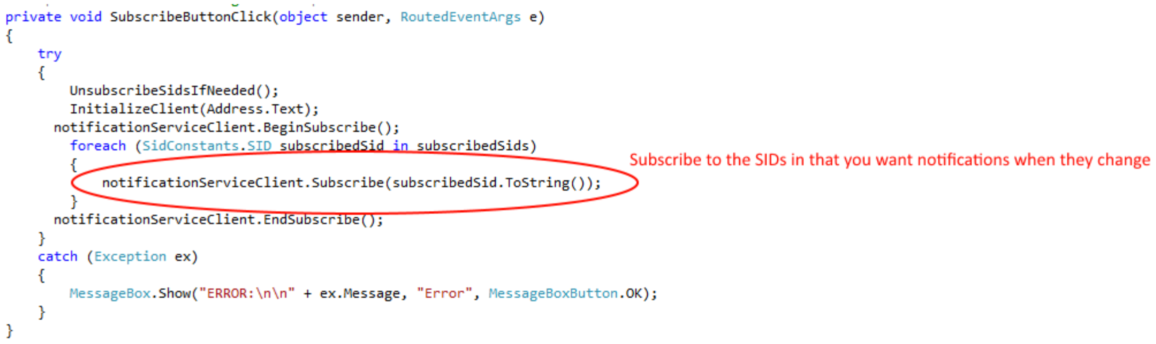
   Your Vendor ID and Password are issued by Hurco



```
////turn on security
notificationServiceBinding.Security.Mode= SecurityMode.Message;
///turn on security


notificationServiceClient.ClientCredentials.UserName.UserName = "YourID"; // your VendorID (1234)
notificationServiceClient.ClientCredentials.UserName.Password = "YourPassword";

try
{
  notificationServiceClient.Open();
}
catch (Exception e)
{
  Messages.Text += "Failed to Connect:\n" + e.Message + "\n" + e.StackTrace;
}
```

Set Vendor ID and Password

5. **Subscribe to the SIDs from the list of subscription SIDs**

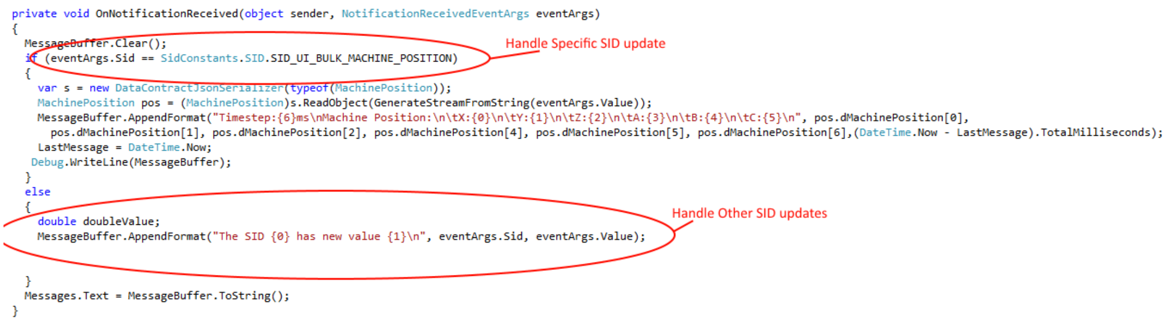


```
private void SubscribeButtonClick(object sender, RoutedEventArgs e)
{
    try
    {
        UnsubscribeSidsIfNeeded();
        InitializeClient(Address.Text);
      notificationServiceClient.BeginSubscribe();
        foreach (SidConstants.SID subscribedSid in subscribedSids)
        {
            notificationServiceClient.Subscribe(subscribedSid.ToString());
        }
      notificationServiceClient.EndSubscribe();
    }
    catch (Exception ex)
    {
        MessageBox.Show("ERROR:\n\n" + ex.Message, "Error", MessageBoxButton.OK);
    }
}
```

Subscribe to the SIDs in that you want notifications when they change

6. **Handle updates to the subscribed SIDs**

   - eventArgs.Sid = name of SID
   - eventArgs.Value = value of SID as string
     - if SID is a bulk will be a JSON string representing the bulk



```
private void OnNotificationReceived(object sender, NotificationReceivedEventArgs eventArgs)
{
  MessageBuffer.Clear();
  if (eventArgs.Sid == SidConstants.SID.SID_UI_BULK_MACHINE_POSITION)
  {
    var s = new DataContractJsonSerializer(typeof(MachinePosition));
    MachinePosition pos = (MachinePosition)s.ReadObject(GenerateStreamFromString(eventArgs.Value));
    MessageBuffer.AppendFormat("Timestep:{6}ms\nMachine Position:\n\tX:{0}\n\tY:{1}\n\tZ:{2}\n\tA:{3}\n\tB:{4}\n\tC:{5}\n", pos.dMachinePosition[0],
      pos.dMachinePosition[1], pos.dMachinePosition[2], pos.dMachinePosition[4], pos.dMachinePosition[5], pos.dMachinePosition[6],(DateTime.Now - LastMessage).TotalMilliseconds);
    LastMessage = DateTime.Now;
   Debug.WriteLine(MessageBuffer);
  }
  else
  {
    double doubleValue;
    MessageBuffer.AppendFormat("The SID {0} has new value {1}\n", eventArgs.Sid, eventArgs.Value);


  }
  Messages.Text = MessageBuffer.ToString();
}
```

Handle Specific SID update

Handle Other SID updates