**Frequently Asked Questions (FAQs)**

**C++ MTConnect Agent**

A list of the most frequently asked questions and their answers is contained in this document to assist you in the development or modification of a MTConnect Agent.

[Question: How can I monitor DataItem changes with an Observer pattern? 1](#_Toc302374852)

[Question: Can I use a http CNC back-end adapter instead of SHDR? 2](#_Toc302374853)

[Question: Is there a simple way in Windows XP/Vista/7 in which to test the HTTP C++ Agent Post interface? 3](#_Toc302374854)

[Question: Is there a simple way to upload an “asset” XML file to the HTTP C++ Agent? 3](#_Toc302374855)

[Question: How do I read my information from the agent configuration file using dlib? 4](#_Toc302374856)

[Question: Can I add my own XML DataItem from within the Agent and not in the devices xml file? 4](#_Toc302374857)

[Question: How do I access the latest value for a given DataItem? 6](#_Toc302374858)

[Question: How can I incorporate other XML schemas and data into MTConnect? 6](#_Toc302374859)

[Question: Is there a way to transmit multiline SHDR data to an MTConnect Agent, which would be very useful in transmitting an XML asset file? 9](#_Toc302374860)

[Question: How do I configure the Agent logger? 10](#_Toc302374861)

[Question: How do I save an Asset file within the Cpp Agent code? 10](#_Toc302374862)

[Question: Can I change the Date Timestamps from UTC to more readable Locl time? 11](#_Toc302374863)

[Question: How can I incorporate windows features into the Cpp Agent? 11](#_Toc302374864)

[Question: How can I incorporate COM features and COM security into the Cpp Agent? 11](#_Toc302374865)

[Question: How can I incorporate another adapter with its own thread into the Cpp Agent? 12](#_Toc302374866)

# Question: How can I monitor DataItem changes with an Observer pattern?

**Answer:** CppAgent implements the observer pattern in which an object maintains a list of observers, and notifies them automatically of any state changes, by calling one of their methods. The observer pattern is based on the signal software of the dlib library.

To implement a DataItem observer in software you need to derive a class from the ChangeObserver class and override the virtual method, signal(), for example:

class BusyObserver : public ChangeObserver

{

public:

BusyObserver(tstring name, DataItem \* item) : \_id(name), \_item(item){}

void signal()

{

Logger<< FATAL << \_id << "Updated " << nowtimestamp() << std::endl;

}

tstring \_id;

DataItem \* \_item;

};

You can write one superclass to derive from ChangeObserver or have per data item class definitions to handle data changes. Once you have the class you need to register it with the actual DataItem so that it can notify an instance of the observer class that a change of state has occurred.

Device \*dev = config.getAgent()->getDeviceByName(devicename);

DataItem \*di = dev->getDeviceDataItem("Xabs");

BusyObserver \* observer = new BusyObserver("Xabs", di)

di->addObserver(observer);

To attach an observer, you will need to access the agent, which contains a list of devices, and inside each device is a list of data items, from which the data item is retrieved. Once you have the data item, you create an new instance of your observer class, and add it as an observer to the data item. There is a corresponding ability to remove the observer also:

di->removeObserver(observer);

Note: ChangeObserver method signal() was made virtual to all override.

virtual void signal() { mSignal.signal(); }

Posted: John Michaloski Tue 08/09/11 12:11:38 PM

# Question: Can I use a http CNC back-end adapter instead of SHDR?

**Answer:** Internally, CppAgent implements an http “post/put” listener for Clients to read samples/events/conditions but also has support for updating DataItems. The POST HTTP method is used when the client needs to send data to the server as part of the web page request.

Here is an example of writing Xabs and Yabs values to the Mazak1 device.

<http://129.6.72.44/Mazak1/sample?Xabs=10.0&Yabs=20.0>

Note, if you use the http post method to update MTConnect data, you will not need a SHDR adapters, and it can be blank within the config file, i.e., agent.cfg:

Adapters

{

}

Posted: John Michaloski Tue 07/26/11 03:01:07 PM

# Question: Is there a simple way in Windows XP/Vista/7 in which to test the HTTP C++ Agent Post interface?

**Answer:** You can use VBScript to update data. Below is a VB Script test to write writing Xabs and Yabs values of the Mazak1 device to the MTConnect Agent.

Function HTTPPost(sUrl, sRequest)

set oHTTP = CreateObject("Microsoft.XMLHTTP")

oHTTP.open "POST", sUrl,false

oHTTP.setRequestHeader "Content-Type", "application/x-www-form-urlencoded"

oHTTP.setRequestHeader "Content-Length", Len(sRequest)

oHTTP.send sRequest

HTTPPost = oHTTP.responseText

Wscript.echo "Status: " & oHTTP.statusText

Wscript.echo "Response: " & oHTTP.responseText

End Function

HTTPPost "http://129.6.72.44/Mazak1/sample?Xabs=10.0&Yabs=20.0", ""

Posted: John Michaloski Tue 07/26/11 10:02:07 AM

# Question: Is there a simple way to upload an “asset” XML file to the HTTP C++ Agent?

**Answer:** Below is a VB Script test to write qmrcylinderplan.xml file to the MTConnect Agent.

Function HTTPPost(sUrl, sRequest)

set oHTTP = CreateObject("Microsoft.XMLHTTP")

oHTTP.open "POST", sUrl,false

oHTTP.setRequestHeader "Content-Type", "application/x-www-form-urlencoded"

oHTTP.setRequestHeader "Content-Length", Len(sRequest)

oHTTP.send sRequest

HTTPPost = oHTTP.responseText

End Function

Dim sData, currentDirectory, filename, devicename

filename ="qmrcylinderplan.xml"

devicename="BORE\_1232?type=Part"

Dim objFSO, objFile, xmlhttp

Const ForReading = 1, ForWriting = 2, ForAppending = 8

' Read file from current directory

currentDirectory = left(WScript.ScriptFullName,(Len(WScript.ScriptFullName))-(len(WScript.ScriptName)))

Set objFSO = CreateObject("Scripting.FileSystemObject")

Set objFile = objFSO.OpenTextFile(currentDirectory + filename, ForReading)

sData = objFile.ReadAll

set xmlhttp = Createobject("MSXML2.XMLHTTP.3.0")

strURL = "http://localhost/asset/" + devicename

xmlhttp.Open "PUT", strURL, false

xmlhttp.Send sData

Wscript.echo "Status: " & xmlhttp.statusText

Wscript.echo "Response: " & xmlhttp.responseText

set xmlhttp=Nothing

Posted: John Michaloski Tue 07/26/11 03:01:07 PM

# Question: How do I read my information from the agent configuration file using dlib?

**Answer:** Assuming the agent has added the ability to determine the current configuration file name, then it is quite easy. First you will need to include the dlib config\_reader.h file, and then open the configuration file as an istream, and then load it into a config reader kernel (main\_cr.load\_from(fin)). Once read you can read a “block” (main\_cr.block("MachineMonitor")) and determine if a key exists(cr.is\_key\_defined("MotionTags")) and then read the keys value (cr["MotionTags"]).

#include <dlib/config\_reader.h>

AgentConfiguration \* \_config;

. . .

ifstream fin(\_config->configfile().c\_str());

config\_reader::kernel\_1a main\_cr;

main\_cr.load\_from(fin);

if (!main\_cr.is\_block\_defined("MachineMonitor")) // is block in config file?

return;

const config\_reader::kernel\_1a& cr = main\_cr.block("MachineMonitor"); // read block

if (cr.is\_key\_defined("MotionTags")) // read a tag in the block

{

std::out << cr["MotionTags"]; // do something

}

Wed 08/10/11 09:32:50 AM

More info at <http://dlib.net/config_reader_ex.cpp.html>

# Question: Can I add my own XML DataItem from within the Agent and not in the devices xml file?

**Answer:** It is possible.

First you will need to modify the Devices.xml or probe.xml file in memory which describes the Device configuration. The web site <http://xmlsoft.org/examples> has many example on how to use libxml2 to navigate, create, delete nodes from the XML tree, and how to do XPath queries. In the case below, we create a path query "//Devices/Device/DataItems" of the node in the XML tree that we want. We then associate a namespace to the path (xmlXPathRegisterNs ) before we execute the Xpath search (xmlXPathEvalExpression).

xmlDocPtr mDoc=\_config->getAgent()->mXmlParser->mDoc;

//http://xmlsoft.org/examples/xpath1.c

xmlXPathContextPtr xpathCtx = NULL;

xmlXPathObjectPtr controllerdataitems = NULL;

std::string path = "//Devices/Device/DataItems";

xpathCtx = xmlXPathNewContext(mDoc);

xmlNodePtr root = xmlDocGetRootElement(mDoc);

if (root->ns != NULL)

{

path = addNamespace(path, "m");

xmlXPathRegisterNs(xpathCtx, BAD\_CAST "m", root->ns->href);

}

controllerdataitems = xmlXPathEvalExpression(BAD\_CAST path.c\_str(), xpathCtx);

if(controllerdataitems == NULL)

{

xmlXPathFreeContext(xpathCtx);

return;

}

Once we have a pointer to the XML node we can can insert a new DataItem in the DataItems list. This is done by creating a new Xml node ( ) and then associating the attributes we want with the node for the equivalent to:

<DataItem category="EVENT" id="id10223" name="probe\_toolchange" type="OTHER" />

This insures that an http agent/current query will return our new data item.

xmlNodeSetPtr nodeset = controllerdataitems->nodesetval;

if(nodeset->nodeMax<1)

return;

xmlNodePtr node = xmlNewChild(nodeset->nodeTab[0], NULL, BAD\_CAST "DataItem", NULL);

xmlNewProp(node, BAD\_CAST "category", BAD\_CAST "EVENT");

xmlNewProp(node, BAD\_CAST "name", BAD\_CAST "probe\_toolchange");

xmlNewProp(node, BAD\_CAST "id", BAD\_CAST "probe\_toolchange");

xmlNewProp(node, BAD\_CAST "type", BAD\_CAST "OTHER");

Next you will need to modify add the DataItem to the device. This entails manually setting up the XML attributes, creating a new DataItem, and then registering the DataItem with both the Device and the Component – in the case below, they are the same entity.

Device \*device = \_config->getAgent()->getDeviceByName(\_devicename);

if(device==NULL)

return;

// Create availability data item and add it to the device.

std::map<tstring,tstring> attrs;

attrs["type"] = "OTHER";

attrs["id"] = "probe\_toolchange";

attrs["name"] = "probe\_toolchange";

attrs["category"] = "EVENT";

// Create new data item

DataItem \*di = new DataItem(attrs);

di->setComponent(\*device);

device->addDataItem(\*di);

device->addDeviceDataItem(\*di);

tstring time = getCurrentTime(GMT\_UV\_SEC);

\_config->getAgent()->addToBuffer(di, "FALSE", time);

Posted: John Michaloski Wed 08/10/11 09:43:39 AM

# Question: How do I access the latest value for a given DataItem?

**Answer:** Data is stored in the agent as a series of ComponentEvents in Checkpoint containers. If you access the “latest” checkpoint container, and use a filter with the id of the data you want, you can retrieve the data as follows:

std::string GetValue(std::string name)

{

Device \*device = \_config->getAgent()->getDeviceByName(“devicename”);

DataItem \* di=device->getDeviceDataItem(name);

if(di==NULL) return "";

Checkpoint mLatest=\_config->getAgent()->mLatest;

std::set<string> aFilter;

aFilter.insert(di->getId());

std::vector<ComponentEventPtr> events;

mLatest.getComponentEvents(events, &aFilter);

if(events.size() < 1)

return "";

return events[0]->getValue();

}

You will need access to the device you are scanning the data for. This does not handle conditions only simple data types, e.g., samples and events. This could also be simplified if you have the id instead of the name.

Posted: John Michaloski Wed 08/10/11 01:15:48 PM

# Question: How can I incorporate other XML schemas and data into MTConnect?

**Answer:** *If you can generate QMR XML, then you 90% of the way there.*

MTConnect version 1.2 has the ability to incorporate and transport independent XML data. The MTConnect specificiton Version 1.2 added assets, (the first case deals with representing CuttingTool (see part 4 of the 1.2 specification.) Assets allows the use of the MTConnect agent as a limited key/value store (<http://en.wikipedia.org/wiki/Associative_array>) with the ability to collect and report entire XML documents as they change within applications.

We will use Quality Measurement Results (QMR) XML Schema to develop the steps involved in communicating the XML documents. The following steps show how this is done.

1. First, you create an XML file like the partial one based on the QMR schema ([qifspecs.org](http://qifspecs.org/)):

<Part timestamp="2011-07-25T13:55:22" assetId="BORE\_1232">

  <Inspection> <!-- this is the start of the QMR specification -->

    <MeasurementResults>

...

    </MeasurementResults>

  </Inspection>

</Part>

The Part is an extension to MTConnect and should be in its own namespace. We could also call it WorkPiece or something else, but we'll use Part for this example.

1. Next you'll need to get a version of the MTConnect Agent (1.2 branch).
2. You will also need to configure the agent. It needs to have this line in the cfg file: AllowPut = true

Here's an example:

Devices = VMC-3Axis.xml

**AllowPut = true**

Adapters {

VMC-3Axis {

Host = localhost

}

}

# Logger Configuration

logger\_config

{

logging\_level = debug

output = cout

}

Next you will need to post data to the agent. See FAQ: “Is there a simple way to upload an “asset” XML file to the HTTP C++ Agent?” for one example on how to upload an XML file. Below, the type=Part bit is because we can store many different asset types and we need to track assets by type.

**http://localhost:5000/asset/BORE\_1232?type=Part**

PUT asset/BORE\_1232?type=Part HTTP/1.1

Host: localhost:5000

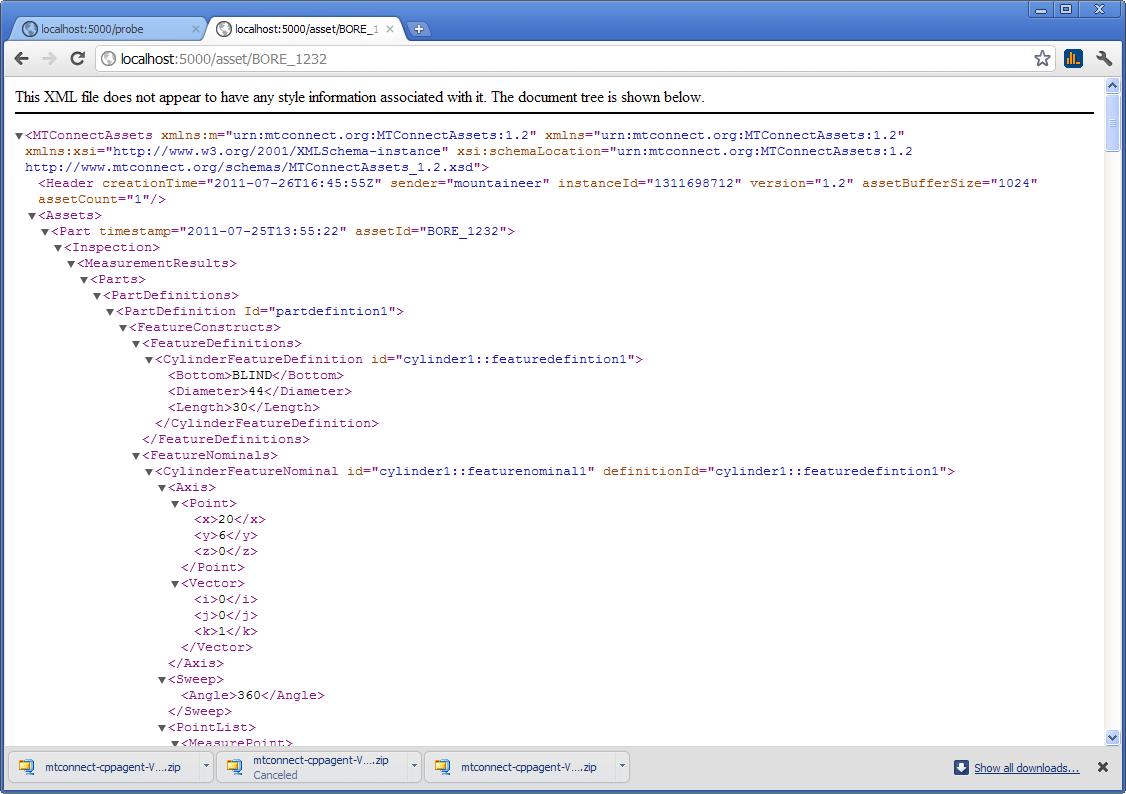
Content-Type: text/plain;charset=UTF-8

Content-Length: <length of file>

[bore.xml Data]

The asset id (BORE\_1232) at the end MUST match the asset Id in the document. That's it for the changes to MTConnect. The new Assets framework can contain any content.

The full document from MTConnect agent after this is done will look like this:



You will also receive an asset changed event telling you a part has been added/changed within the /current MTConnect query (highlighted in bold below):

<?xml version="1.0" encoding="UTF-8"?>

<MTConnectStreams xmlns:m="urn:mtconnect.org:MTConnectStreams:1.2" xmlns="urn:mtconnect.org:MTConnectStreams:1.2" xmlns:xsi="<http://www.w3.org/2001/XMLSchema-instance>" xsi:schemaLocation="urn:mtconnect.org:MTConnectStreams:1.2 <http://www.mtconnect.org/schemas/MTConnectStreams_1.2.xsd>">

  <Header creationTime="2011-07-25T19:44:18Z" sender="flori-2.local" instanceId="1311622912" version="1.2" bufferSize="131072" nextSequence="47" firstSequence="1" lastSequence="46"/>

  <Streams>

    <DeviceStream name="VMC-3Axis" uuid="000">

      <ComponentStream component="Device" name="VMC-3Axis" componentId="dev">

        <Events>

          <Availability dataItemId="avail" timestamp="2011-07-25T19:41:52.147505Z" sequence="11">UNAVAILABLE</Availability>

**<AssetChanged dataItemId="dev\_asset\_chg" timestamp="2011-07-25T19:42:16.855924Z" sequence="46" assetType="Part">BORE\_1232</AssetChanged>**

        </Events>

      </ComponentStream>

    </DeviceStream>

  </Streams>

</MTConnectStreams>

The application monitors the assetchanged events and then grabs the data. This is the same process as with cutting tools. There's more docs on this in the latest MTC 1.2 docs regarding asset storage and such. We also have asset counts in the header as well:

  <Header creationTime="2011-07-25T19:46:41Z" sender="flori-2.local" instanceId="1311622912" version="1.2" bufferSize="131072">

    <AssetCounts>

      <AssetCount assetType="Part">1</AssetCount>

    </AssetCounts>

  </Header>

Note: these changes are still in beta and will be finalized with when the docs have finished review.

Posted: John Michaloski Fri 08/12/11 04:05:29 PM

# Question: Is there a way to transmit multiline SHDR data to an MTConnect Agent, which would be very useful in transmitting an XML asset file?

**Answer:** There is also a new “multiline” extension to MTConnect using SHDR which allows you to transmit multiple lines at a time. Below is and EBNF representation of the SHDR with the new multiline asset implementation:

<SHDR> ::= <Date> "|" <StatementList>

<StatementList> ::= <Statement> | <Statement> EOL <StatementList>

<Statement> ::= <SimpleStatement> | < MultilineStatement >

<SimpleStatement> ::= <Tag> "|" <Value> { "|" <Value>}\*

<MultilineStatement> ::= "@" <Tag> "@" "|" ID "|" --multiline—{A-Z}+ .\* --multiline—{A-Z}+

where beginning and ending multiline statement “--multiline—{A-Z}” must match.

Here is an example implementing the Asset SHDR:

2011-07-25T13:55:22|@ASSET@|BORE\_1232|Part|--multiline--AAAA

<Part timestamp="2011-07-25T13:55:22" assetId="BORE\_1232">

  <Inspection>

    <MeasurementResults>

...

  </Inspection>

</Part>

--multiline--AAAA

The multiline will scan until it sees a matching --multiline-XXXX token at the beginning of the line. The contents will act similar to the post.

Posted Mon 08/15/11 09:58:48 AM

# Question: How do I configure the Agent logger?

**Answer:** The agent.cfg is responsible for the Agent configuration. Inside the agent.cfg file, logging configuration is specified using the logger\_config block.

|  |  |
| --- | --- |
| **logger\_config configuration items** | |
| logger\_config | The logging configuration section. |
| logging\_level | The logging level: trace, debug, info, warn, error, or fatal.  Default:info |
| output | The output file or stream. If a file is specified specify as:  “file <filename>”. cout and cerr can be used to specify the standard output and standard error streams. Defaults to the same directory as the executable.  Default: file adapter.log |

You can change the logging\_level to specify the verbosity of the logging as well as the destination of the logging output.

1. logger\_config

2. {

3. logging\_level = debug

4. output = file debug.log

5. }

This will log everything from debug to fatal to the file debug.log. For only fatal errors you can specify the following:

1. logger\_config

2. {

3. logging\_level = fatal

4. }

The default file is agent.log in the same directory as the agent.exe file resides. The default logging level is info. To have the agent log to the command window:

1. logger\_config

2. {

3. logging\_level = debug

4. output = cout

5. }

This will log debug level messages to the current console window. When the agent is run with debug, it is sets the logging configuration to debug and outputs to the standard output as specified above.

# Question: How do I save an Asset file within the Cpp Agent code?

**Answer:** The agent has a method addAsset which can be used to save an Asset (XML code) in a ring buffer. The procedure needs pointers to the Agent, Device and then the Asset ID (BORE\_1232), Type(XML Head element) and a Body, i.e., the xml file. Saving the Asset code, for the DataItem BORE\_1232 is shown below, where \_config is the Agent Configuration.

// <DataItem type="ASSET\_CHANGED" id="BORE\_1232" category="EVENT" name="BORE\_1232"/>

std::string aBody = "<Part><Inspection><MeasurementResults/></Inspection></Part>";

std::string aId="BORE\_1232";

std::string type="Part";

Device \*device = \_config->getAgent()->getDeviceByName(\_device);

\_config->getAgent()->addAsset(device, aId, aBody, type);

Posted: John Michaloski Wed 08/17/11 09:48:07 AM

# Question: Can I change the Date Timestamps from UTC to more readable Locl time?

**Answer:** The agent Xmlprinter.cpp has the code:

THROW\_IF\_XML2\_ERROR(xmlTextWriterWriteAttribute(writer, BAD\_CAST "creationTime", BAD\_CAST getCurrentTime(GMT).c\_str()));

Posted Mon 08/29/11 10:58:58 AM

# Question: How can I incorporate windows features into the Cpp Agent?

**Answer:** Inside the cppagent.cpp file you can add the windows subsystem (to run instead of main()), which is shown in the following code so that it will use WinMain as the primary main.

#pragma comment(linker, "/SUBSYSTEM:WINDOWS")

int APIENTRY WinMain(HINSTANCE hInstance,

HINSTANCE hPrevInstance,

LPTSTR lpCmdLine,

int nCmdShow)

You do not need to remove tmain or main. They will be ignored.

Using WinMain, argc/argv are gone but you can use \_\_arc, \_\_argv instead:

config.main( \_\_argc,(const char \*\*) \_\_argv );

Posted: John Michaloski Mon 08/29/11 10:58:58 AM

# Question: How can I incorporate COM features and COM security into the Cpp Agent?

**Answer:** First, add the windows subsystem described earlier and then add the COM functionality you require, as illustrated below:

int APIENTRY WinMain(HINSTANCE hInstance,

HINSTANCE hPrevInstance,

LPTSTR lpCmdLine,

int nCmdShow)

{

HRESULT hr;

hr = ::CoInitialize(NULL);

hr = ::CoInitializeSecurity( NULL, //Points to security descriptor

-1, //Count of entries in asAuthSvc

NULL, //Array of names to register

NULL, //Reserved for future use

RPC\_C\_AUTHN\_LEVEL\_NONE, //The default authentication //level for proxies

RPC\_C\_IMP\_LEVEL\_IDENTIFY, //The default impersonation //level for proxies

NULL, //Reserved; must be set to NULL

EOAC\_NONE, //Additional client or //server-side capabilities

NULL //Reserved for future use

);   
. . .

Posted: John Michaloski Mon 08/29/11 10:58:58 AM

# Question: How can I incorporate another adapter with its own thread into the Cpp Agent?

**Answer:** You will need to derive (subclass the class AgentConfiguration) and override the start and stop methods. Then, you can create a new adapter, and spawn off a thread to run your own adaper.

class AgentConfigurationEx : public AgentConfiguration

{

public:

AgentConfigurationEx() {}

MyAdapter \* myadapter;

virtual void start()

{

sLogger << LENTRY << "MTConnect Agent Service Started " << nowtimestamp() << "\n";

\_myadapter = new MyAdapter ((AgentConfiguration \*) this);

myadapter->Start() ;

}

// Start the core server. This blocks until the server stops!

AgentConfiguration::start();

}

virtual void stop() { … }

. . .

};

You will then need to rewrite the main() routine to use the new agent configuration class.

int main(int aArgc, const char \*aArgv[])

{

AgentConfigurationEx config;

. . .

config.main(aArgc, (const char \*\*) aArgv);

Posted: John Michaloski Mon 08/29/11 10:58:58 AM

# Question: Is there a simple way in C++ in which to communicate to the the HTTP Agent Post interface?

**Answer:** Asio is a cross-platform C++ library for network and low-level I/O programming that provides developers with a consistent asynchronous model using a modern C++ approach. There is a boost and non-boost implementation. Below shows a SendHttp function to send any szrequest to an http server.

#include "boost/asio.hpp"

using namespace std;

Bool SendHttp(string szrequest, string domainname="127.0.0.1", string port="80")

{

boost::asio::ip::tcp::iostream stream;

stream.connect(domainname, port);

stream << "GET " << szrequest << " HTTP/1.0\r\n"

<< "\r\n"

<< std::flush;

if( stream.bad())

return false;

std::string response\_line;

while(!stream.bad() && !std::getline(stream, response\_line).eof())

{

stdout << response\_line.c\_str();

}

return true;

}

Here is an example of using SendHttp to write Xabs and Yabs values to the Mazak1 device.

<http://129.6.72.44/Mazak1/sample?Xabs=10.0&Yabs=20.0>

SendHttp("http://129.6.72.44/Mazak1/sample?Xabs=10.0&Yabs=20.0", "129.6.72.44");

Posted: John Michaloski Fri 09/23/11 11:47:41 AM

# Question: In an adapter, how can I do Microsoft COM communication, without the importing a DLL?

**Answer:** You will need the name of the Com component program id (PROGID) or the CLSID. Below shows how to do it with the program id.

CComPtr<IDispatch> \_appdispatch;

...

\_appdispatch.CoCreateInstance(L"PCDLRN.Application");

Then, the COM component has to be implemented as an automation IDispatch interface so that you can do method and property name lookup via the Dispatch interface. It’s easiest if you use ATL GetPropertyByName method and the default implementation of CComPtr<IDispatch>.

template<typename VariantType>

\_variant\_t GetTypedProperty(CComPtr<IDispatch> pDispatch, \_bstr\_t propname, VariantType defaultval)

{

\_variant\_t var(defaultval);

if(pDispatch==NULL) return var;

HRESULT hr= pDispatch.GetPropertyByName(propname, (VARIANT\*)&var);

// could throw exception if failed...

return var;

}

void FakeAdapter::gatherDeviceData()

{

USES\_CONVERSION;

HRESULT hr;

mAvailability.available();

\_variant\_t version, pProg, speed;

version = GetTypedProperty<BSTR>(\_appdispatch, L"VersionString", L"");

pProg = GetTypedProperty<IDispatch \*>(\_appdispatch, L"ActivePartProgram", NULL);

// assume works

CComPtr<IDispatch> \_progdispatch = (IDispatch \*) pProg;

speed=GetTypedProperty<LONG>(\_progdispatch, L"Speed", 0);

Don’t forget to call ::CoInitialize(NULL) or everything will fail.

Posted: John Michaloski Tue 10/11/11 01:16:28 PM

# Question: Can I implement a simple MTConnect agent using PCDMIS?

**Answer:** PCDMIS supplies a COM interface (OLE Automation) that make VB programming easy. If we stick to using the Properties whenever possible (as opposed to the event callbacks) we don’t even need to have the DLL to program it in C++, instead we can use the .

|  |  |
| --- | --- |
| **POWER** | Unclear. |
| **PROGRAM** | ApplicationObject.ActivePartProgram |
| **CONTROLLER\_MODE** | ApplicationObject.OperatorMode, TRUE if in operator mode (MANUAL), FALSE=AUTOMATIC |
| **EXECUTION** | Running if (ApplicationObject.ActivePartProgram.Speed > 0) |
| **LINE** | N/A. |
| **FAULT** |  |