**How to Accept Very Large Returns via MeF
Platform: .net framework**

**Purpose**: The purpose of this document is to explain how to accept the largest returns possible using a MeF gateway based on the .net framework.

**Disclaimer:** It should be noted that some of the techniques described here be used with discretion, as they could result in the default .net “out of memory” exception handling not taking effect. This could result in your server using up its physical memory to the point where it becomes unstable.

**Audience**: Techies

**Techniques:**

1. Use the .net framework 4.5 compiled as a 64-bit application using the gcAllowVeryLargeObjects runtime flag in the app.config.
	1. The gcAllowVeryLargeObjects flag allows .net 4.5 to ask the operating system for more than 2GB of memory to store an array. It is necessary only if your state wishes to receive zipped returns > 2GB. Note that the IRS master zip attachment itself is sent over SOAP/MTOM as a base-64 byte array.
	2. Reference: <http://msdn.microsoft.com/en-us/library/hh285054.aspx>
	3. Disclaimer: Using this flag may result in the default .net “out of memory” exception not firing
	4. Useful Blog: <http://bhrnjica.net/2012/07/22/with-net-4-5-10-years-memory-limit-of-2-gb-is-over/>

Example:

<configuration>
 <runtime>
 <gcAllowVeryLargeObjects enabled="true" />
 </runtime>
 <startup>
 <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.5" />
 </startup>
</configuration>

1. Make sure you have the maximum amount of memory possible on your server
	1. The maximum memory a Windows Server 2008 R2 Standard x64 edition can utilize is 32GB. The Datacenter, Enterprise, and HPC server x64 editions can all utilize up to 2TB.
		1. Reference: [http://msdn.microsoft.com/en-us/library/windows/desktop/aa366778(v=vs.85).aspx](http://msdn.microsoft.com/en-us/library/windows/desktop/aa366778%28v%3Dvs.85%29.aspx)
2. If you are using another .net framework on a 32-bit based server, you will not be able to accept returns >2GB. But you can maximize your applications total memory heap potential by:
	1. Using the Visual Studio command line utility editbin to make your .net application largeaddressaware.
		1. Reference: <http://msdn.microsoft.com/en-us/library/d25ddyfc.aspx>

Example:



* 1. To use this you must also set a /3GB flag in your Boot.ini and reboot the server
		1. Reference: [http://www.maxi-pedia.com/3GB+switch+Windows+boot.ini+3+GB](http://www.maxi-pedia.com/3GB%2Bswitch%2BWindows%2Bboot.ini%2B3%2BGB)
			1. Note: This article is important to read in order to understand how a 32-bit Windows operating system allocates memory to your processes.
		2. Important: DO NOT TRY THIS ON WINDOWS XP. It may cause Windows XP to cease booting
1. Set your app.config WSE 3 Settings to allow for larger return sizes
	1. ExecutionTimeoutInSeconds – Season to Taste!
	2. MaxMessageLength
		1. Set to -1 to allow for unlimited SOAP responses
			1. Disclaimer: Setting this to -1 may result in .net’s default “out of memory” exception handling not firing
		2. Set to a particular byte size to receive an “out of memory” memory once the limit is reached. Note that this is in bytes:
			1. 1024 = 1 KB
			2. 1048576 = 1 Megabyte (or 1024 \* 1000)
			3. 262144000 = 250 Megabyte (or 1048576 \* 250)

Example 1 (30 Minutes with Unlimited File Size):

      <microsoft.web.services3>

             <messaging>

                    <mtom clientMode="On"/>

                    <executionTimeoutInSeconds value="1800"/>

                    <maxMessageLength value="-1"/>

             </messaging>

            ...

</microsoft.web.services3>

Example 2 (2 Hours with 250MB limit):

<microsoft.web.services3>

             <messaging>

                    <mtom clientMode="On"/>

                    <executionTimeoutInSeconds value="7200"/>

                    <maxMessageLength value="262144000"/>

             </messaging>

            ...

</microsoft.web.services3>

1. Ask your network people if they have any timeouts set on your proxy server themselves.
	1. Note: Regardless of the timeouts you set on the client, a proxy server can override these if it wants to.
	2. Reference: <http://httpd.apache.org/docs/2.2/mod/mod_proxy.html>
2. Set any timeouts on your classes that inherit from WSE 3 WebServicesClientProtocol to allow for larger return sizes
	1. Note: This timeout is in milliseconds. You can calculate milliseconds using the .net TimeSpan class.

Sample Code (See following pages):

1. A class that sets timeout & proxy information
	* Note: Inherits from WebServicesClientProtocolExtensionBase

Imports System.Configuration

Imports System.net

Public MustInherit Class FedStateCommonServiceBase

    Inherits WebServicesClientProtocolExtensionBase

    Public Sub New()

        MyBase.New()

        '

        'Set timeout

        '

        Dim minutes As Double

        minutes = ConfigurationManager.AppSettings.Get("MeF\_Timeout\_Minutes")

        Dim timeSpan As New TimeSpan(0, minutes, 0)

        Me.Timeout = timeSpan.TotalMilliseconds

        '

        'Set Proxy

        '

        Dim proxyPath As String = ""

        proxyPath = ConfigurationManager.AppSettings.Get("Proxy")

        Dim proxy As WebProxy = New WebProxy(proxyPath)

        'Set credentials if configured to

        Dim proxyUser As String = ""

        Dim proxyPassword As String = ""

        proxyUser = ConfigurationManager.AppSettings.Get("ProxyUser")

        proxyPassword = ConfigurationManager.AppSettings.Get("ProxyPassword")

        Me.Proxy = proxy

    End Sub

End Class

1. A class that sets HTTP.KeepAlive and turns MTOM off when some IRS webservice calls return non-MTOM based responses
	* Note: Inherits from WSE 3’s WebServicesClientProtocol

Imports Microsoft.Web.Services3

Imports System.Net

<System.ComponentModel.DesignerCategoryAttribute("code")> \_

Public MustInherit Class WebServicesClientProtocolExtensionBase

    Inherits WebServicesClientProtocol

    Protected Overrides Function GetWebRequest(ByVal uri As System.Uri) As System.Net.WebRequest

        Dim WebReq As WebRequest = MyBase.GetWebRequest(uri)

        Dim prop As System.Reflection.PropertyInfo

        prop = WebReq.GetType().GetProperty("Request")

        Dim HttpReq As HttpWebRequest

        HttpReq = CType(WebReq, HttpWebRequest)

        If Not HttpReq Is Nothing Then

            HttpReq.KeepAlive = False

            HttpReq.ProtocolVersion = HttpVersion.Version11

        End If

        Return WebReq

    End Function

    Protected Overrides Function GetWebResponse(ByVal request As System.Net.WebRequest) As System.Net.WebResponse

        Dim Response As WebResponse = MyBase.GetWebResponse(request)

        If Response.Headers(HttpResponseHeader.ContentType).ToLower.StartsWith("text/xml") Then

            Me.RequireMtom = False

        End If

        Return Response

    End Function

End Class

1. A WSDL Class that was modified to inherit from the FedStateCommonServiceBase class that sets the timeout and proxy, which in turn inherits from the WebServicesClientProtocolExtensionBase class that sets the HTTP.KeepAlive and toggles MTOM based on SOAP responses, which in turn inherits from WSE 3’s WebServicesClientProtocol

Imports System

Imports System.ComponentModel

Imports System.Diagnostics

Imports System.Web.Services

Imports System.Web.Services.Protocols

Imports System.Xml.Serialization

<System.CodeDom.Compiler.GeneratedCodeAttribute("wsdl", "2.0.50727.3038"), \_

System.Diagnostics.DebuggerStepThroughAttribute(), \_

System.ComponentModel.DesignerCategoryAttribute("code"), \_

System.Web.Services.WebServiceBindingAttribute(Name:="Login", [Namespace]:="<http://www.irs.gov/a2a/mef/MeFMSIServices>")> \_

Partial Public Class Login

    Inherits FedStateCommonServiceBase

    ...

End Class