DVS “Mock” Environment - Document Guide

(Tech Services Contract 2.3.1 refers)

# Introduction

## Purpose

This document provides details of the USI sample code for integrating with the USI external web service. Sample code delivered as a Visual Studio 2019, .Net 4.8 C# application.

## Audience

Developers and/or system integrators should read this document.

## Overview

The USI Sample Code demonstrates how a Student Management System can integrate with the USI Agency web services. The web services provide following functions:

1. Create a USI record for an individual and receive an immediate response;
2. Verify a USI for an individual and receive an immediate response;
3. Submit a batch of USI creation requests, and receive a receipt number;
4. Retrieve the results of a previous batch creation request via the receipt number; and
5. Perform USI verification for a batch of requests and receive an immediate response.
6. Update contact details for a given USI
7. Get a list of Non DVS documents for the organisation code set to 0002 by default

# Prerequisites

## Registration

To use the USI web service RTO’s must be registered and current on Training.Gov.Au (TGA). Other VET related organisations can register with the USI Taskforce. Registration will provide organisations with an Organisation code (which is provided when calling the USI web services).

## Digital Certificate

Callers of the USI web services will require a Machine to Machine credential (M2M) matching the ABN registered with TGA in order to verify their identity. For ThirdParty environments, device credentials are issued by ATO in the form of an Xml document (named KeyStore-USI.Xml).

USI web services are secured using the ATO Machine Authentication Service (MAS). The MAS is a service used to validate the credential and generate security tokens. The security tokens are used to secure one or more messages sent to the USI web services.

The sample code provided demonstrates how to extract the certificate from the KeyStore and make the call to MAS.

## Development Environment

The sample code has the following dependencies:

* Visual Studio 2019 (Solution file format);
* Microsoft .Net platform version 4.6.1
* KeyStore Manager kit (AKM), sourced from the Australian Taxation Office (SBR);
* IIS 7.0 or higher;
* Bouncy Castle Crypto component;
* Enterprise Library Common Logging component.

# Using The Sample Code

## Environment

New release of the USI Sample Code integrates with the ATO ThirdParty environment. The ThirdParty environment provides the following endpoints:

The endpoint URL for the ATO MAS is: <https://softwareauthorisations.acc.ato.gov.au/R3.0/S007v1.3/Service.svc>.

The endpoint URL for the USI web service calls (test environment) is:

<https://3pt.portal.usi.gov.au/Service/UsiService.svc>

PLEASE NOTE: Once you are ready to deploy your software to your production environment, the production endpoints need to be used.

The production endpoint URL for the ATO MAS is:

<https://softwareauthorisations.ato.gov.au/R3.0/S007v1.3/service.svc>

The production endpoint URL for the USI web service is:

<https://portal.usi.gov.au/Service/UsiService.svc>

## Command Line Interface

The sample code comes in the form of a command line driven console application to exercise each of the USI web service functions. The console application has the following command line arguments:

|  |  |
| --- | --- |
| Argument | Description |
| /c | Creates a single USI record, and immediately verifies the record (makes two distinct USI web service calls). |
| /b | Uploads a batch of USI applications for processing, and returns a receipt number. |
| /r | Retreives the processing status for a batch of applications using the receipt number. |
| /v | Performs a verification for a batch of USI applications. |
| /uc | Updates the contact details for the given inputted USI |

## Source Files

The following table provides an overview of each of the files included in the USISampleCode solution:

| File | Description |
| --- | --- |
| Program.cs | Contains the Main() startup method and each of the methods used to call the USI web services. |
| RequestFactory.cs | Creates request objects for the USI web service calls. |
| ServiceChannel.cs | Extracts the certificate from the KeyStore, creates a WCF channel for USI web service calls, and configures the channel to use the M2M certificate when getting a security token from ATO MAS. |
| App.config | Application configuration file containing the WCF configuration for the following:   * Calling the ATO MAS (Bindings, Endpoints and Behaviours) * Calling the USI web service (Bindings, Endpoints and Behaviours) * WCF trace messages (client side) |
| Abr.AuskeyManager.dll | AUSkey ADK component used to access the local Keystore. This file is acquired from SBR. |
| BouncyCastle.CryptoExt.dll | AUSkey ADK dependency. This file is acquired from SBR. |
| Common.Logging.dll | AUSkey ADK dependency. This file is acquired from SBR. |
| Common.Logging.Log4net.dll | AUSkey ADK dependency. This file is acquired from SBR. |
| Log4net.dll | AUSkey ADK dependency. This file is acquired from SBR. |

## Basic Flow

The following is a basic outline of the processing flow:

1. Attach an error handler for the application;
2. Set the call back handler function for validating a certificate (only for debug);
3. Parse the command line arguments to determine which USI web service call to invoke;
4. Create a WCF channel to the USI web service;
5. Extract the certificate from the keystore using the AKM;
6. Configure the WCF channel with the M2M certificate;
7. Open the WCF channel;
8. Invoke the USI web service call; and
9. Write the response to stdout.

## Required Changes

The following customisations are required in the following files before using the Sample Code:

### Dependencies

The AUSkey AKM will need to be sourced from SBR and added to the sample code solution. USI Office does not redistribute the AKM with the sample code.

The following assemblies must be added to the Dependencies folder within the sample code folder structure. Once added, Visual Studio will resolve the dependencies. These assemblies are found in the AUSkey AKM:

* Abr.AuskeyManager.dll
* BouncyCastle.CryptoExt.dll
* Common.Logging.dll
* Common.Logging.Log4Net.dll
* Log4net.dll

### ServiceChannel.cs

The const string Alias needs to match the credential identifier as found in the Keystore.xml file. The credential identifier is in the form of “ABRP:99999999999\_99999999”.

The const string Password needs to match the keystore/credential password.

|  |
| --- |
| // AusKey Key Id.  const string AUSkey\_KeyAlias = "ABRP:12300002581\_10080080";  const string AUSkey\_Password = "Password1!"; |

### RequestFactory.cs

The const string OrganisationCode must be changed to “VA1802” for service providers. RTO’s or HE providers should use “VA1803”.

|  |
| --- |
| // Issued organisation code.  const string OrganisationCode = “VA1803”; |

## Document VerIfication

Document verification in the ATO Third Party environment uses a mock service (the DVS service is not called). Valid DVS document types include:

* Citizenship;
* Registration by Descent;
* Driver License;
* Medicare Card;
* Passport (Australian);
* Non-Australian Passport (with Australian Visa); and
* Birth Certificate.

By default the sample code sets the DVSCheckRequired property to true on the ApplicationType object for each request. As a result, the USI web service will invoke the mock Document Verification Service to verify the DVS document.

Each document type uses a key value for verification. The sample code provides a comment highlighting which fields are used for the key values. Note these key values are only appropriate for the USI web services in the Third Party environment.

|  |
| --- |
| public static DVSDocumentType Medicare(){  return new MedicareDocumentType(){  CardColour = MedicareDocumentTypeCardColour.Green,  ExpiryDate = "2015-12",  IndividualRefNumber = "3",  MedicareCardNumber = "1111111111" // Ensures the mock DVS service will pass the verification.  };  } |

The following table outlines the acceptable key values that will result in a pass or unavailable status for the document check.

| Document Type | Key Field | Value | Result |
| --- | --- | --- | --- |
| Citizenship | Stock Number | ACC111111 | Pass |
| Citizenship | Stock Number | ACC000000 | Unavailable |
| Registration by Descent | AcquisitionDate | 01/01/2013 | Pass |
| Registration by Descent | AcquisitionDate | 12/12/2012 | Unavailable |
| Driver Licence | Licence Number  Card Number | 111111  A001234567 | Pass |
| Driver Licence | Licence Number | 000000 | Unavailable |
| Medicare Card | Card Number | 1111111111 | Pass |
| Medicare Card | Card Number | 0000000000 | Unavailable |
| Passport (Australian) | Document Number | X1111111 | Pass |
| Passport (Australian) | Document Number | X0000000 | Unavailable |
| Non-Australian Passport (with Australian Visa) | Passport Number | 111111 | Pass |
| Non-Australian Passport (with Australian Visa) | Passport Number | 000000 | Unavailable |
| Birth Certificate | Registration Number | See separate table for details | |
| Birth Certificate | Registration Number | See separate table for details | |

Setting the value to any other will result in a failed request.

Please refrain from submitting “Unavailable” values to the BulkUpload operation, as these will cause the corresponding applications to get stuck in the system indefinitely.