Registry Interfaces for BPC open source development

Contact: terryg@intechiq.com

Phone: (+61) 2 8305 2112

Document: Interface Description

Status/Rev: 1.3

Date: 06 July 2021

Table of Contents

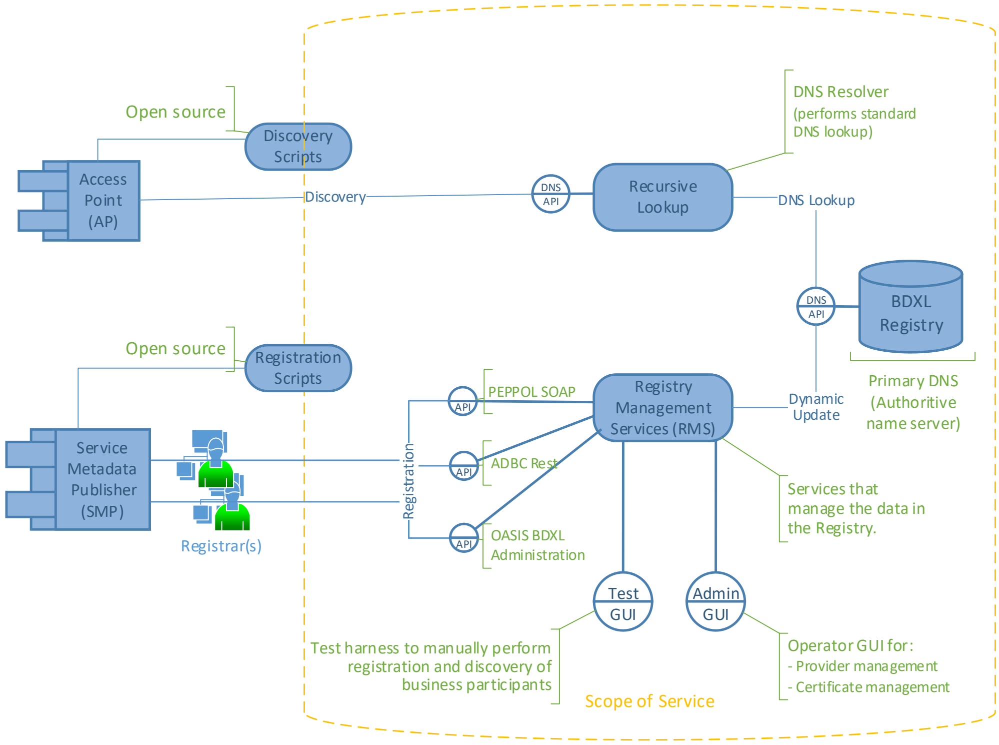
1 Introduction

The Digital Locator Service (DLS) provides the components necessary to manage Service Providers and Business Participants in an e-invoicing network.

1.1 About this document

This document is intended to be used by Registry Operators and Service Providers to develop solutions to register and manage Service Providers and Business Participants respectively.

1.2 Solution Scope and Context


Open source
Discovery Scripts
Access Point (AP)
Discovery
DNS API
Recursive Lookup
DNS Resolver (performs standard DNS lookup)
DNS Lookup
DNS API
BDXL Registry
Primary DNS (Authoritive name server)
Open source
Registration Scripts
Service Metadata Publisher (SMP)
Registrar(s)
Registration
API
PEPPOL SOAP
API
ADBC Rest
API
OASIS BDXL Administration
Registry Management Services (RMS)
Dynamic Update
Services that manage the data in the Registry.
Test GUI
Admin GUI
Test harness to manually perform registration and discovery of business participants
Operator GUI for:
- Provider management
- Certificate management
Scope of Service

This document describes the following interfaces and data structures:

1. Registry Management Service interfaces including
   a. SML SOAP – an implementation of SOAP SML Interface
   b. ADBC Rest – an implementation of the Australian Digital Business Councils (ADBC) Rest interface for managing an e-invoicing registry
   c. OASIS BDXL Administration Service – this is a place holder indicating the intention to implement this interface when its specification becomes available
2. Open-Source libraries – a collection of open source libraries for:
   a. Business Participant discovery
   b. Business Participant registration and management
3. Test Harness GUI (open source) – a sample application that illustrates how perform registration and discovery of a Business Participant in the Registry.
4. Administration GUI – an Administration Interface to manage Service Providers and their certificates. This interface also allows for audit logging features.

2 DEFINITIONS

The following terms are used through this document:

The following is a list of **non-technical** terms used throughout this document:

Term	Description
Business Participant (BP) or Participant	An entity, typically a business, that wishes to be able to send and receive electronic invoices.
Service Metadata Publisher (SMP) or Digital Service Provider (DSP) or Digital Capability Publisher (DCP)	A data service that provides a sending access point with the digital capabilities that are used to exchange digital documents in an e-invoicing network.
Registry	A DNS system that stores the records of Business Participants identifiers and the end point of their SMPs that manages e-invoicing transactions on their behalf.
Registration or Register	The process of recording a Business Participant in a Registry.
Registrar	An entity who performs Registrations for Business Participants using a Registration Service.
BP Discovery	The process used by an Access Point to find the end-point URL of an SMPs service that is providing an e-invoices service for the BP.
Registration Service (RS)	A software service that performs a Registration.
Registry Management Service (RMS)	A software service that performs a Registration and other functions related to maintenance of a registry.
Service Metadata Locator (SML)	A Registration Service
Digital Locator Service (DLS)	A Registration Service
Registration Service Operator (RSO)	An entity who manages a Registration Service
SML Operator	An entity who manages a SML
DLS Operator	An entity who manages a DLS
Registration Service Developer (RSD)	An entity who develops the Registration Service software.
Registration Service Interface (RSI)	The programmatic interface of a Registration Service.
e-invoicing network	A collection of policies, rules and technical infrastructure that is provided to allow the exchange of electronic invoicing.
e-invoicing network operator	A governing body that oversees the running of an e-invoicing network.
Service Provider	An entity that provides a component or service in an e-invoicing network.
Schema	A Business Identifier schema type and schema value that represents the type of identifier of a Business Participant. Eg: *participantIdentifierMetaScheme + participantIdentifierSchemeValue*
Access Point	A service used to exchange digital documents. Access Points lookup the digital addresses a DCP used by a Business Participants (see BP Discovery).

The following is a list of **technical** terms used throughout this document:

participantIdentifier or Participant Identifier	Participant identifier – The fully qualified identifier including **participantIdentifierMetaScheme**, **participantIdentifierSchemeValue** and **participantIdentifierIdValue**	urn:oasis:names:tc:ebcore:partyid-type:iso6523:9908::810418052
participantIdentifierMetaScheme	Participant identifier - the Meta Scheme type	urn:oasis:names:tc:ebcore:partyid-type:iso6523
participantIdentifierSchemeValue	Participant identifier - value of the Meta Scheme	0037, 0088, 0151, 9908
participantIdentifierValue	Participant identifier - value of the identifier and its meta scheme value	0037:22501236
participantIdentifierIdValue	Participant identifier just the number/id on its own	22501236
ServiceMetadataPublisherID	The issued unique Id of the SMP	396 397
universalParticipantId	Not available for this trial	iso6523-actorid-upis::9908:810418052
urnParticipantId		urn:oasis:names:tc:ebcore:partyid-type:iso6523:9908::810418052
fullIdentifier	Either the universalParticipantId or the urnParticipantId, whichever is agreed by an implementation but a combination of **participantIdentifierScheme** and **participantIdentifier**	
serviceInterfaceList	A comma limited list containing the meta interface implemented by the service.	List items can be from the following: SML2 SML2.1 DLS2
registryNameSpace	The root domain name space that contains the Business Participant and Service Provider records representing the entities involved in the digital framework	bpc.bdxl.services
registryType	The structure of the records contained in the Registry identified by the registryNameSpace	• bdxl1Root • bdxl1RootSubDirScheme • bdxl1SubDirScheme • peppol1CName peppol1Naptr
metaSchemeType	The format of the Participant Identifier.	• iso6523-actorid-upis 1. urn:oasis:names:tc:ebcore:partyid-type

3 Registry Management APIs

This section describes the Application Programming Interfaces available to manage a Registry.

The intention is for the developer of a Service Provider to use one of these interfaces at their choice. Multiple interfaces are provided to support Service Providers who may have existing integration with other e-invoicing networks, or technical constraints in regards to the services they can access.

For example, a PEPPOL Service Providers may have existing integration with the PEPPOL SML SOAP interface which they can utilise here, alternatively a Service Provider may have a preference to use a REST Interface and would then use the ADBC REST interface. The OASIS BDXR are planning to create a (new) BDXL Administration Service, and this will also be supported when available.

3.1 SML SOAP Interface

This is an implementation of Connecting Europe Facilities (CEF) SML Interface Specification which is part of the PEPPOL e-Delivery network (Please refer to https://ec.europa.eu/cefdigital/wiki/download/attachments/82773331/%28eDelivery%29%28SML%29%28ICD%29%281.07%29.pdf?version=1&modificationDate=1538386436086&api=v2).

Note it does not use the CEF SML software – it is an interface implemented in the DLS Software.

All APIs operate via Mutual TLS communication via https protocols. Using invalid client certificates or not providing a client certificate with an API request will be rejected prior to reaching the application.

This service is available (to holders of a valid certificate) here:

https://sml1.dev.bpc.bdxl.services/api/v1/

<Note: only participant Create and Delete functions exposed as at 2021-03-22- – other functions being added>

3.2 ADBC Rest Interface

This interface is based on the Australian Digital Business Councils (ADBC) Rest interface for managing an e-invoicing registry as described in http://digitalbusinesscouncil.com.au/digital-capability-locator/. This REST API supports JSON and XML requests.

This service is available (to holders of a valid certificate) here:

https://dls1.dev.bpc.bdxl.services/dlssml/services/

3.3 BDXL Registry Administration Interface

Placeholder for when a specification for this interface is available.

4 Open-Source Components for Developers

Open-source items are provided for developers to integrate Registration and Discovery processes into software used by SMPs and Access Points respectively.

These items are:

1. Business Participant Management (Registration) functions –
2. Business Participant Discovery function
3. GUI application which demonstrates the functionality of the above by providing an application which allows manual Registration and Discovery of Business Participants.

4.1 Business Participant Management (Registration) Functions

<Documented separately for now - TBA. >

4.2 Business Participant Discovery Functions

4.2.1 Discover Business Participant (*lookupParticipant*)

4.2.1.1 Narrative

Input:

The Business Participant's Participant Identifier.

The root location (domain name space) of the Registry

registryType - the type of Registry (optional, default 'bdxl1Root')

metaSchemeType – the type of participantIdentifierMetaScheme used in the Registry (optional, default taken from Participant Identifier).

Output

A structure containing a return code, the DNS record and the Service Name and Address of the digital service for the SMP which services the Business Participant

Functionality:

The library implements the hashing to convert the Participant Identifier to the DNS record structure and then performs the DNS lookup to retrieve the result from DNS.

4.2.1.2 C# function description

Namespace : com.intechiq.dls

Class Library : registryDiscovery

Class: registry

Properties:

public string registryNameSpace	The root domain name space that contains the Business Participant and Service Provider records representing the entities involved in the digital framework. Eg: dev.bpc.bdxl.services
public string dnsServer	The server ip address or dns alias of a dns server (optionally including the port) that is able to resolve queries for the registryNameSpace
public string registryType	One of these types of known registry 'standards': • bdxl1Root • bdxl1RootSubDirScheme • bdxl1SubDirScheme • peppol1CName • peppol1Naptr
public string metaSchemeType	The structure of the Participant Identifier. Handled structures are: • iso6523-actorid-upis • urn:oasis:names:tc:ebcore:partyid-type:iso6523

Structures

struct participantRecord

public string returnCode	'0' for success, or an error number or code returned from the lookup participant method
public string returnString	the full DNS record or error string.
Public string providerServiceName	The type of service, eg Meta:SMP
Public string providerServiceAddress	A URL or IP Address containing the end point of the provider's digital service.

struct participant

public string identifierIdValue	See definitions
public string identifierMetaScheme	See definitions
Public string identifierSchemeValue	See definitions
Public string fullIdentifier	See definitions

Constructors:

To look-up a participant, an object of the registry class has to be created, identifying the name space (identifying the location) of the Registry.

- public registry(string registryNameSpace)
- public registry (*string* registryNameSpace, *string* dnsServer[1])
- public registry (*string* registryNameSpace, *string* dnsServer, string registryType)
- public registry (*string* registryNameSpace, *string* dnsServer, string registryType, string metaSchemeType)

  On construction, these Properties are set as follows:

string registryNameSpace	This must be set by the calling application. Eg: "dev.bpc.bdxl.services"
string dnsServer	This can be passed in by the calling application, or the constructor will set this to the default primary dns identified from the system. (identified by System.Net.DnsEndPoint Class)
string registryType	Unless explicitly set, this is set to the default of: "bdxl1Root" *<Note: possible enhancement – discover from the Registry what type it is>*
string metaSchemeType	Unless explicitly set, this is set left empty.

[1] Example set from client: ns1dev.bpc.bdxl.services

Methods (of class registry)

Method : lookupParticipantRecord

Access Modifier : public

Return type : participantRecord

Parameters : (string participantIdentifier)

: (string participantIdentifier, string dnsServer)

: (string participantIdentifier, string dnsServer, string registryType, string metaSchemeType)

Summary This method calls the computeHash method to convert the Participant Identifier into its Hash equivalent that is stored in the Registry. It then performs a DNS lookup to retrieve the DNS record for this Business Identifier, and then extracts information from the DNS record to populate the return structure.

Example Registry myRegistry1 = new Registry(‘dev.bpc.bdxl.services’, ‘ns1dev.bpc.bdxl.services’)

structParticipantRecord = myRegistry1.lookupParticipantRecord("urn:oasis:names:tc:ebcore:partyid-type:iso6523:0037::4035811992070")

Result: structParticipantRecord.returnString: 60 IN NAPTR 100 10 "U" "Meta:SMP" "!^.*$!http://www.smp5.com!"

structParticipantRecord.serviceAddress: http://www.smp5.com

Methods (of class registry)

Method : computeHash

Access Modifier : public

Return type : *string*

Parameters : (string participantIdentifier)

: (string participantIdentifier, string registryType)

: (string participantIdentifier, string registryType, string metaSchemeType)

Summary This method:

1. Calls the parseParticipantIdentifier (string participantIdentifier, <string metaSchemeType>) method to identifier the discrete components that are used to build the Hash.
2. Builds the Hash in accordance with the registryType as follows:

   If registryType = ‘bdxl1Root’

   base32(SHA256(lowercase(<participant.fullIdentifier>)))

   If registryType = ‘bdxl1RootSubDirScheme’

   base32(SHA256(lowercase(<participant.fullIdentifier>))) + ‘.’ + base32(SHA256(lowercase(<participant.identifierMetaScheme>:<participant.identifierSchemeValue>))

   If registryType = ‘bdxl1SubDirScheme’

   base32(SHA256(lowercase(<participant.identifierIdValue>))) + ‘.’ + base32(SHA256(lowercase(<participant.identifierMetaScheme>:<participant.identifierSchemeValue>))If registryType = ‘peppol1CName’

   ‘b-‘ + base32(MD5(lowercase(<participant.identifierMetaScheme>:<participant.identifierIdValue>)))

   If registryType = ‘peppol1Naptr’

   base32(SHA256(lowercase(<participant.identifierMetaScheme>:<participant.identifierIdValue>)))

Example Registry myRegistry1 = new Registry(‘dev.bpc.bdxl.services’, ‘ns1dev.bpc.bdxl.services’)

strMyParticipantHash = myRegistry1.computeHash ("urn:oasis:names:tc:ebcore:partyid-type:iso6523:0037::4035811992070")

Result: ISMVASTUDQD3GRHO2DNPDQBQKWM3BWLBRJ7XDQKXU6IAPT2H6MZA

→(assumes default registryType [‘bdxl1Root’] and metaSchemeType [‘urn:oasis:names:tc:ebcore:partyid-type:iso6523’])

Method : parseParticipantIdentifier

Access Modifier : private

Return type : *struct participant*

Parameters : (string participantIdentifier)

: (string participantIdentifier, string metaSchemeType)

Summary This method identifies the discrete components that are used to build the Hash, and returns those components as items of the participant struct as follows:

Participant item	**Value**
identifierIdValue	Extract from ParticipantIdentifier. Assume it is the right most string in the list where the participantIdentifier is considered a list of strings separated by : or ::. Valid examples (non-exhaustive) 12345678, 9876543212
identifierSchemeValue	Extract from ParticipantIdentifier Valid examples (non-exhaustive): 0037, 0088, 0151, 9908 Assume it is the second right most string in the list where the participantIdentifier is considered a list of strings separated by : or ::.
identifierMetaScheme	If metaSchemeType.IsNullOrWhiteSpace then extract the participantIdentifierMetaScheme from the participantIdentifier. Assume it is the remaining (left part) of the string, trimmed of : after the participantIdentifierIdValue and participantIdentifierSchemeValue are extracted. Else set to metaSchemeType Valid examples (non-exhaustive): iso6523-actorid-upis urn:oasis:names:tc:ebcore:partyid-type:iso6523
fullIdentifier	If metaSchemeType.IsNullOrWhiteSpace then participantIdentifier else if metaSchemeType like ‘urn*’ (<identifierMetaScheme>: <identifierSchemeValue>**::** <identifierIdValue> if metaSchemeType like ‘*upis’ (<identifierMetaScheme>**::** <identifierSchemeValue>: <identifierIdValue> else participantIdentifier

Example Registry myRegistry1 = new Registry(‘dev.bpc.bdxl.services’, ‘ns1dev.bpc.bdxl.services’)

structMyParticipant = myRegistry1. parseParticipantIdentifier ("urn:oasis:names:tc:ebcore:partyid-type:iso6523:0037::4035811992070")

Result: ISMVASTUDQD3GRHO2DNPDQBQKWM3BWLBRJ7XDQKXU6IAPT2H6MZA

structMyParticipant.participantIdentifierIdValue = 4035811992070

structMyParticipant.participantIdentifierSchemeValue = 0037

structMyParticipant.participantIdentifierMetaScheme =
urn:oasis:names:tc:ebcore:partyid-type:iso6523

5 Graphical User Interfaces

5.1 Overview

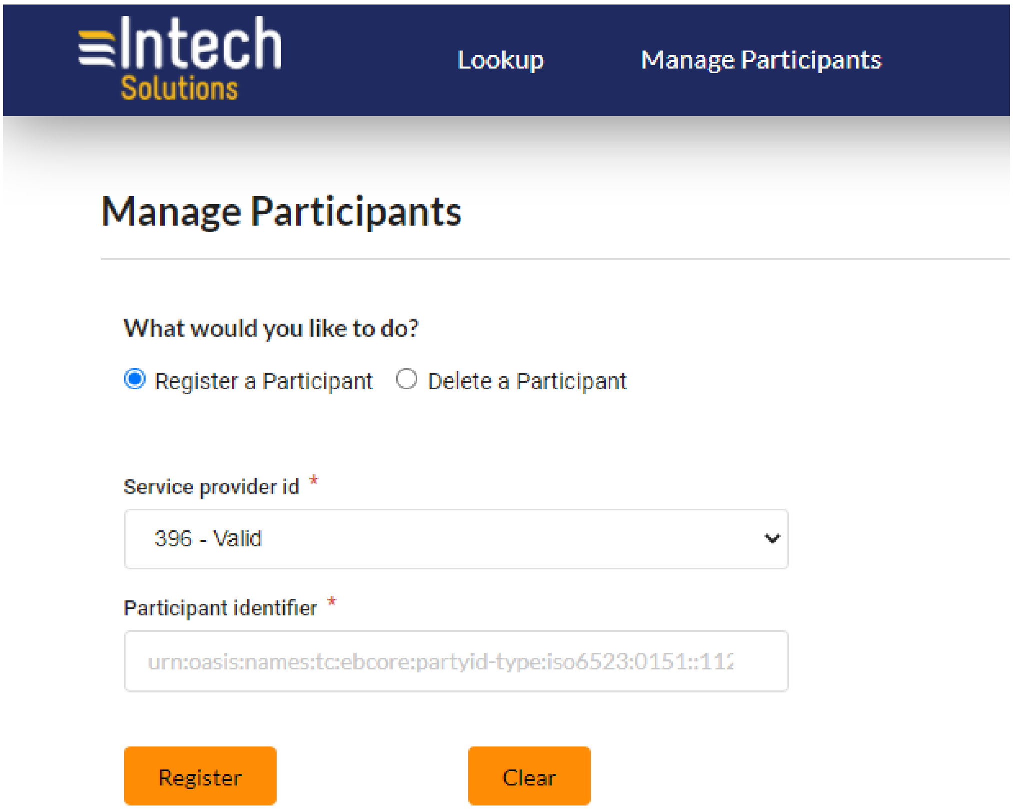
The follows client applications are available:

1. Sample integration for SMP's and Access Points (C#.Net Web GUI) – an application which illustrates the functionality used by SMPs and Access Points to register a Participant in the Registry and discover/lookup a Participants digital service providers address.
2. Registry Management GUI – an interface for a Registry Operator to manage their records in the Registry. This provides functions to register, monitor and manage SMP's and their use of the Registration Service.

5.2 Sample integration for SMP (C#.Net Web GUI)

A sample has been built as both a console application described below and as a hosted web application available via the following URL:

https://th.dev.bpc.bdxl.services/testharness/



Manage Participants

What would you like to do?

● Register a Participant ○ Delete a Participant

Service provider id *

396 - Valid

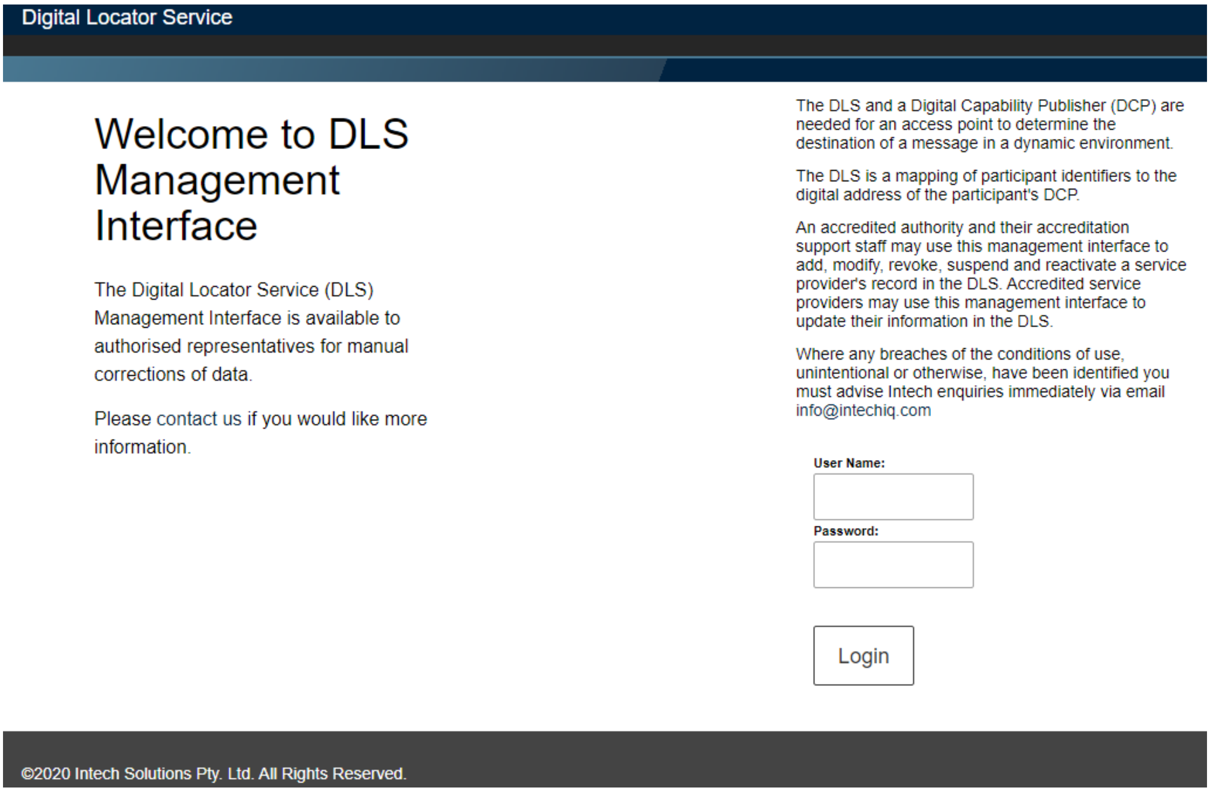
Participant identifier *

urn:oasis:names:tc:ebcore:partyid-type:iso6523:0151::112

These can be provided as source code.

5.3 OPERATOR MANAGEMENT GUI

https://mgmt.dev.bpc.bdxl.services/mgmtgui/pages/home



Welcome to DLS Management Interface

The Digital Locator Service (DLS) Management Interface is available to authorised representatives for manual corrections of data.

Please contact us if you would like more information.

The DLS and a Digital Capability Publisher (DCP) are needed for an access point to determine the destination of a message in a dynamic environment.

The DLS is a mapping of participant identifiers to the digital address of the participant's DCP.

An accredited authority and their accreditation support staff may use this management interface to add, modify, revoke, suspend and reactivate a service provider's record in the DLS. Accredited service providers may use this management interface to update their information in the DLS.

Where any breaches of the conditions of use, unintentional or otherwise, have been identified you must advise Intech enquiries immediately via email info@intechiq.com

Configured User Credentials:

<please contact us for credentials>