Right to Lift

Usage of Right to Lift Data Exchange Standards for

Downstream Europe Road Transport Fuels Movements



Document Properties

|  |  |
| --- | --- |
| Version | 1.0 |
| Last Update | 20-March-2013 |
| Prepared by | Bryan McPadden/Mark Garton |
| Owner | PIDX International |
| Contact Information | [Bryan.McPadden@telventdtn.com](mailto:Bryan.McPadden@telventdtn.com)  [Mark.Garton@telventdtn.com](mailto:Mark.Garton@telventdtn.com) |

Disclaimer

**This document is intended to capture field usage detail about the Right To Lift Data Exchange Standard for Oil and Gas Downstream Road Transport Fuels Movements in Europe.**

**The document is open to all workgroup participants and interested parties to comment on.**

Contents

[1 Introduction 4](#_Toc342333881)

[1.1 Purpose of the document 4](#_Toc342333882)

[1.2 Glossary 4](#_Toc342333883)

[1.3 PIDX Right To Lift Command Overview 5](#_Toc342333884)

[1.3.1 PIDX Authorization Request 6](#_Toc342333885)

[1.3.2 PIDX Authorization Response 7](#_Toc342333886)

[1.3.3 PIDXBOL 8](#_Toc342333887)

[1.3.4 PIDXBOLResponse 9](#_Toc342333888)

[2 Detailed Element Data Population Recommendations 10](#_Toc342333889)

[2.1 Scenarios using the LoadingType field: 11](#_Toc342333890)

[2.2 Multiple Parties in the AuthParties structure 12](#_Toc342333891)

[2.2.1 One Party Example 12](#_Toc342333892)

[2.2.2 Two Party Example 13](#_Toc342333893)

[2.3 Cancelled Loads – No Load BOLs 13](#_Toc342333894)

# Introduction

## Purpose of the document

This document will show examples and give recommendations on how to implement the PIDX SOAP Web Service utilizing the published WSDL from PIDX.

The intended audience is:

1. Any TAS Vendor planning to implement Right to Lift Protocol.
2. Any potential Clearing House service provider.
3. Any relevant trade association.

## Glossary

|  |
| --- |
| **Allocation** – Defines a control over amount of product that can be lifted against a contract over a specific period.  **SOAP** –originally defined as **Simple Object Access Protocol**, is a protocol specification for exchanging structured information in the implementation of Web Services in computer networks. It relies on XML Information Set for its message format, and usually relies on other Application Layer protocols for message negotiation and transmission. PIDX has chosen to use HTTPS as the Application layer protocol for messaging between the DCH/TAS vendors.  **WSDL -** The **Web Services Description Language** is an XML-based interface description language that is used for describing the functionality offered by a web service. A WSDL description of a web service (also referred to as a *WSDL file*) provides a machine-readable description of how the service can be called, what parameters it expects, and what data structures it returns. It thus serves a purpose that corresponds roughly to that of a method signature in a programming language. |
| **Clearing House (CH)/DCH** – External company selected to deliver a facility for the exchange of data between oil companies and to manage this service. |
| **Lifting** – Loading of one or more products.  **Lifting Controls** – The ability to authorize or refuse a load based on a customer’s position against their credit limit and/or allocation in real time.  **Nomination** – Scheduled primary transport trip. May be for a loading or discharging at an OSP. |
| **OSP** – Outside Supply Point. Oil depot from a company other than the Contract Owner. Typically another oil company or terminal operator.  **PIDX** – Petroleum Industry Data Exchange. The American Petroleum Institute’s (API) committee on Electronic Business Standards and Processes. |
| **TAS** – Terminal Automation System. System to authorize and automate truck loadings. |
| **XML** – Extensible Markup Language, is a simplified subset of the Standard Generalized Markup Language (SGML, ISO 8879), which provides a file format for representing data, a schema for describing data structure, and a mechanism for extending an annotation HML with semantic information. |

## General Recommended Coding Approach

It is common practice to generate code from a WSDL which simplifies the implementation of generating and receiving messages. This document will explain a how this is accomplished through some commonly available Java tools, as well as .NET tools. If you use a standard compiler, generally the Enterprise edition of the complier software will provide this functionality.

**Server/Clearing House Implementation Notes**

There are generally two different ways you can implement a SOAP web service. The approaches are often referred to as a TOP DOWN or BOTTOM UP approach. Top Down approach uses the WSDL to generate server side code, while bottom up approach uses the XSD to generate a WSDL. For RTL, it’s HIGHLY recommended that the TOP DOWN approach is used to remain 100% compatible with the published WSDL that TAS vendors will be coding against. The BOTTOM UP approach can often generate slight difference in the XML contract which will cause the service to be incompatible with the PIDX standards. See the implementation guidelines published by Microsoft for further information on this recommendation (<http://msdn.microsoft.com/en-us/library/vstudio/hh339625(v=vs.100).aspx#fsadrttyyty>).

.

**Client Side Implementation Notes**

The theory behind publishing a standard WSDL by the PIDX committee was to allow each DCH (Data Clearing House) to be 100% compatible with the standard and remove from scope any custom coding for each DCH. The TAS Vendor should NOT have to know which DCH they are connecting to, they should be able to connect to PIDX compliant vendors without any code modifications. The Examples contained in the document are Java and Windows WFC examples, but they should be extendable to any other programming language.

## PIDX Right To Lift Server/DCH Example

### Creating Windows Service

Required Software:

* Visual Studio 2010/2012

To generate the service contract:

* Open Visual Studio command prompt
* Type: svcutil /mc "{WSDL file path}"
* If your WSDL has any supporting XSD files pass those in as arguments after the WSDL.
  + svcutil /mc "{WSDL file path}" "{XSD 1 file path}" "{XSD 2 file path}"...

This will generate a contract interface. You will need to add it to your project and create a service that implements that interface.

To setup the service host:

* Set the endpoint to use wsHttpBinding.
* Set the security mode to “Transport”
  + This is done by using the bindings element in the config file.

### Creating Windows Client using ScvUtil.exe

Required Software:

* Visual Studio 2010/2012

To generate the service contract:

* Open Visual Studio command prompt
* Type: svcutil \*.wsdl \*.xsd /language:C# (Or VB)

This will create the proxy from the wsdl and xsd’s.

Client setup:

* Set WS-Addressing to true.

For message level security the clientCredentials object will need to be populated with a username and password.

## Java Examples

Example of how to create Server/Client side code for Java is described in the document below. Various readily available tools are referenced in the document and may need to be downloaded to setup an environment. The documentation her is provided as a means to get going, and it is presumed that the reader has some expertise in Java and these tools. Additional research on the tools maybe required and is left to the developer to do this research.

Reference Docs in this directory

**Create a web service from a WSDL using Jax.doc**

**Data referenced in Document above is in the right\_to\_lift\_new directory.**