Equifax Gateway - Reference Documentation

Version: A1.0.0

Table of Contents

1 Introduction

The Equifax Gateway is made up of a number of services, which consist of a number of operations. The services currently supported are given below.

The **Consumer Service** provides the facility to conduct a number of different operations which relate to different types of consumer searches (e.g. credit search, identity verification search), as well as the ability to supply new Alias or Associate data to Equifax.

The **Address Matching Service** provides a standalone consumer address matching facility and an address lookup via postcode facility.

The **Security Service** facilitates the functions of logon and changing client secret.

This reference documentation describes the various operations available via the supported services and how to interact with them to execute service enquiries, once access to the services has been granted.

1.1 Consumer Service

The request will be generated by the client, including the access token obtained by the logon function, upon receipt of the request, it will be validated. If the validation fails, an error will be returned and the processing will finish. If all validation has been passed a bureau data search will be conducted on Equifax's database and a response in accordance with the details as specified within this guide, will be returned to the client system.

The request message can also be validated for content and structure against the schemas referenced within the published WSDL prior to submission by the client system.

There are a number of similar operations included within the Consumer Service, the key difference being the type of 'search' that is logged at Equifax during enquiry processing. Via these operations either data in its raw state or coded state or a combination of the two can be returned. Also incorporated is access to Risk Navigator, Equifax's generic credit reference score, which makes use of characteristics based on supplied addresses together with the added power of characteristics based on linked address data.

You request the elements of the service you require via data groups, typically one data group per type of data to be returned. The response will contain elements representing those data groups requested, plus a number of elements not included on the request (e.g. supplied address or linked address data). Data will only be returned subject to the restrictions imposed by the Data Protection Act 1998 and membership of the various closed user groups e.g. Insight).

1.2 Address Matching Service

The request will be generated by the client, including the access token obtained by the logon function, upon receipt of the request, it will be validated. If the validation fails, an error will be returned and the processing will finish. If all validation has been passed the address matching operation will be invoked and a response in accordance with the details as specified within this guide will be returned to the client system.

The request message can also be validated for content and structure against the schemas referenced within the published WSDL prior to submission by the client system.

There are two distinct address matching operations included within the Address Service, those being Address List and Address Match.

One of three possible results will be returned; unique match, multiple match or no match.

1.3 Security Service

The **Security Service** facilitates the functions of logon and changing client secret.

The operation is:

- Logon
- Change Client Secret

1.4 Client Testing

When available, clients will be given access to Equifax's Quality Assurance (QA) environment for testing of the service(s) they require. Please discuss the service availability in the QA environment with your Account Manager. All processing will be the same on both the live and the QA systems. Clients will be supplied with a test 'pack' containing names and addresses and associated data present on the QA database.

Subject to prior agreement, a further period for commission testing on the production system may also be permitted. Client enquiries during this period will NOT generate search records at the matched addresses.

The **client secret** associated with the **client id** will be set to expire in accordance with the standard Equifax security policies. This expiry period will be configured at the time the account is created and by default will be set to **30 days**. This will therefore require the client to employ the Change Client Secret operation, periodically, to ensure that the account does not become locked and therefore unusable.

1.5 Versioning

As the services delivered via the Equifax Gateway evolve, WSDL/schema changes will be made; operations and services will be added, updated or removed. This will ultimately require service users to make changes to ensure that they benefit from new data or functionality. However, where possible Equifax will preserve backward compatibility of services through versioning and will communicate new releases of services to users in a timely manner through account management.

1.6 Confidentiality Statement

All information contained in this document is provided in confidence for the sole purpose of adjudication of this document and shall not be used for any other purpose, and shall not be published or disclosed wholly or in part to any other party without Equifax Ltd's prior permission in writing and shall be held in safe custody. These obligations shall not apply to information that is published or becomes known legitimately from some other source than Equifax Ltd.

2 Web Service Messages

2.1 SOAP Message Protocol

The Equifax Gateway comprises a set of discrete SOAP web service endpoints for use by Equifax clients as well as internal Equifax products.

The SOAP interface is not dependent on the hardware platforms or implementation mechanisms of client systems. The same interface is accessible to PCs, mobile devices, mid-range machines and mainframes.

XML SOAP request and response messages are supplied and consumed by client software wishing to interact with the web services and the provided services are described by published web services description language (Web Service Description Language) documents.

SOAP is a standard mechanism for data interchange between computer systems. There are many commercial and open source tools and libraries that can be used to create SOAP requests and parse SOAP responses.

The set of elements, attributes, entities and notations that are used within the SOAP (XML) messages are formally defined in a set of W3C XML Schema Documents (XSDs in the published WSDLs).

2.1.1 Connection Method

SOAP requests are passed to the Equifax web service interface via HTTPS (Hypertext Transfer Protocol over Secure Socket Layer, or HTTP over SSL). This is one of the underlying protocols of the Internet and defines how messages are formatted and securely transmitted.

HTTPS is the native secure communication mechanism for web servers and browsers and again there are a number of commercial and open source libraries and components available to handle this protocol. Equifax web services are not available via unencrypted HTTP.

The SOAP 1.1 protocol being used in the web service interface supports HTTP/1.0 and HTTP/1.1, with the latter being recommended to support all possible 'binding' features. For a complete specification of HTTP/1.0 see HTTP/1.0 Informational RFC 1945. For a complete specification of HTTP/1.1 see Hypertext Transfer Protocol - HTTP/1.1.

HTTP is a stateless protocol (i.e. no connection information is maintained between transactions). Like most network protocols, HTTP uses the client-server model. An HTTP client opens a connection and sends a request message to a HTTP server. The server then returns a response message, usually containing the resource that was requested. After delivering the response, the server closes the connection.

SOAP request messages must be sent to the application via a HTTP POST command and the XML must be URL encoded.

Example SOAP request message for the ping operation

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
                  xmlns:pin="http://ewsconsumer.services.uk.equifax.com/schema/v4/ping/pingrequest">
   <soapenv:Header/>
   <soapenv:Body>
      <pin:pingRequest>
         <clientRef>PING_ENQUIRY</clientRef>
      </pin:pingRequest>
   </soapenv:Body>
</soapenv:Envelope>
```

HTTP POST command containing the ping XML request message

```
HTTP POST command containing the interface ping XML request
POST /xmlii/EWSConsumerService-v4_0/ewsConsumerService/ HTTP/1.1
Accept-Encoding: gzip,deflate
Content-Type: text/xml;charset=UTF-8
SOAPAction: ""
User-Agent: Jakarta Commons-HttpClient/3.1
Host: xmlii.glas01.ntg.equifax.com
Content-Length: 3560

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:v4="http://ewsconsumer.services.uk.equifax.com/schema/v4/ping/pingrequest">
  ..
</soapenv:Envelope>
```

Note

Throughout the rest of this document the term HTTP may be used when referring to the application protocol, however it should be noted that this is only available over SSL (therefore HTTPS) and it should not be assumed that an unencrypted alternative delivery mechanism is available.

This requires a HTTP header as detailed in section 2.2

2.1.2 Physical Connectivity

Whilst HTTP is transport independent, the additional requirement of running HTTP over an SSL protocol limits the network transport protocol to TCP/IP.

The bandwidth requirements of the physical network link will depend on the number of requests/responses being generated and also the response time requirements for each external client/system.

2.1.3 Security

The approach taken to secure the Equifax Gateway web services is to utilise the Open Authentication OAuth 2 Protocol which provides Authorization header block within the HTTP header for attaching access token information in the request.

Since confidentiality and data integrity is enforced on the web service through the use of Transport Layer Security (TLS v1.2) - by sending and receiving messages over HTTPS - no signing or encryption of the supplied user credentials is necessary.

Equifax implemented an internal session management mechanism to allow multiple disparate transactions to be tracked across requests.

When a logon request containing a client id and client secret is received, it is validated and a unique alphanumeric access token is returned within the logon response. This access token needs to be put in HTTP Authorization Header block as Bearer Type to make subsequent calls to consumer and address services. This will allow transactions to be tracked across disparate request messages within the Equifax application architecture.

Once authenticated, use the same token for all subsequent consumer and address service calls for the tokens lifespan. Currently a token is valid for 5 hours (300 minutes) and we recommend a new one is requested just prior to expiry for continuity of service.

2.2 Message HTTP Headers

Each request needs a standard HTTP Authorization header with Bearer token type structure incorporating to OAuth standards.

Bearer token to be put in HTTP POST Request

```
In HTTP Header

Authorization: Bearer <Access Token>
```

If the Access token element is not present in the request, or if authentication fails, a SOAP Fault message will be returned to the requester.

2.3 Fault Messages

If an error is encountered within the web service processing, a SOAP Fault response will be generated (see the fault and error handling section for more details).

> ⚠ If the system is unavailable, either no response or an HTTP error response code will be returned.

Sample SOAP Fault response where credit search request fails authentication

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
   <soapenv:Header/>
   <soapenv:Body>
      <soapenv:Fault>
         <faultcode>soapenv:Server</faultcode>
         <faultstring xml:lang="en">Invalid Token error</faultstring>
         <detail>
            <ns2:creditSearchFault xmlns:ns2="http://ewsconsumer.services.uk.equifax.com/schema/v4">
               <code>EWSC0005</code>
               <message>Invalid Access Token</message>
            </ns2:creditSearchFault>
         </detail>
      </soapenv:Fault>
   </soapenv:Body>
</soapenv:Envelope>
```

3 Getting Started

Once client id and client secret have been obtained to connect to the web services, the published WSDLs describing the services can be used to generate SOAP client applications to interact with each service.

Only the latest version of the WSDLs and the URLs will be supported going forward, all clients who are using a version less than v4.0 will be required to migrate to v4.0 at the earliest opportunity.

Application URLs (the application URL through browser is forbidden):

- Test Application URL for Consumer Services Version 4.0
- Test Application URL for Address Match services Version 4.0
- Test Application URL for Security services Version 4.0
- Production Application URL for Consumer Services Version 4.0
- Production Application URL for Address Match services Version 4.0
- Production Application URL for Security services Version 4.0

All the test and live environment service WSDLs are available as a .zip file at SDKs tabs. Select your preferred service WSDLs .zip file and download.

3.1 Services using Java

The information in this section should assist in the creation of a Java-based (using the JAX-WS reference implementation) web service client with which to interact with the required service(s).

The first step is to run the wsimport tool against the published WSDL for the service(s).

```
wsimport https://api.equifax.co.uk/efx-uk-xmlii-consumer/v1
```

```
wsimport https://api.equifax.co.uk/efx-uk-xmlii-security/v1
```

This will generate a client API model (set of Java classes) that can be used to: build requests, invoke the service and interpret responses with default package names. For example:

```
import com.equifax.uk.services.ewsconsumer.schema.v4.creditsearch.creditsearchrequest.CreditSearchRequest;
import com.equifax.uk.services.ewsconsumer.schema.v4.creditsearch.creditsearchresponse.CreditSearchResponse;
import com.equifax.uk.services.schema.v4.CreditSearchFault;
import com.equifax.uk.services.schema.v4.*;
import com.equifax.uk.services.schema.v4.EWSConsumerService;
import com.equifax.uk.services.schema.v4.EWSConsumerService_Service;
import com.equifax.uk.services.schema.v4.EWSSecurityService;
import com.equifax.uk.services.schema.v4.EWSSecurityService_Service;
```

If the 'target package' option (-p <package name>) is used when generating the client API, name collisions may have to be handled using custom binding extensions within the client configuration.

Create a credit search request object using the API generated for the Consumer service:

```
// create credit search request
CreditSearchRequest creditSearchRequest = new CreditSearchRequest();
creditSearchRequest.setClientRef("SAMPLE_ENQUIRY");

// single applicant enquiry
RequestSoleSearch rss = new RequestSoleSearch();
CreditSearchConfig config = new CreditSearchConfig();
config.setOptIn(true);
rss.setCreditSearchConfig(config);
```

```
// set match criteria
MatchCriteria mc = new MatchCriteria();
mc.setAssociate(MatchCriteriaStatus.NOT_REQUIRED);
mc.setAttributable(MatchCriteriaStatus.NOT_REQUIRED);
mc.setFamily(MatchCriteriaStatus.NOT_REQUIRED);
mc.setPotentialAssociate(MatchCriteriaStatus.NOT_REQUIRED);
mc.setSubject(MatchCriteriaStatus.REQUIRED);
rss.setMatchCriteria(mc);

// set requested data
RequestedData rd = new RequestedData();
OutputAddressRequest oar = new OutputAddressRequest();
oar.setMaxNumber(0);
rd.setOutputAddressRequest(oar);
rss.setRequestedData(rd);

// set up applicant details
RequestPerson person = new RequestPerson();
Name name = new Name();
name.setForename("MOHAMMAD");
name.setSurname("BUTLER");
person.setName(name);

ResidenceInstance residence = new ResidenceInstance();
ResidentialStructuredAddress rsa = new ResidentialStructuredAddress();
rsa.setNumber("26");
rsa.setPostcode("CB6 2JR");
residence.setAddress(rsa);
person.setCurrentAddress(residence);
rss.setPrimary(person);

creditSearchRequest.setSoleSearch(rss);
```

Create a service object using the generated service class. This will be used to invoke the service with the credit search request.

```
EWSConsumerService_Service serviceProxy = new EWSConsumerService_Service();
EWSConsumerService consumerService = serviceProxy.getEWSConsumerServiceSoap11();
```

The service object can then be used to invoke web service operations. For example, to send the creditSearchRequest we have just created to the creditSearch operation:

```
try {
   creditSearchResponse = consumerService.creditSearch(creditSearchRequest);
} catch (CreditSearchFault ex) {
   log.error("Fault returned by consumer service.");
   log.error("Code:    " + ex.getFaultInfo().getCode());
   log.error("Message: " + ex.getFaultInfo().getMessage());
} catch (Exception ex) {
   ex.printStackTrace();
}
```

Complete Java example

A complete example client program built using API classes generated from the published Consumer WSDL is given below:

```
/*
 * Sample client code to interact with the Consumer Service and Security Service following
 * API class generation from the published WSDL using wsimport.
 */
package com.equifax.main;

import java.net.URL;
import java.util.Collections;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import javax.xml.namespace.QName;
import javax.xml.ws.BindingProvider;
import javax.xml.ws.handler.MessageContext;

import com.equifax.uk.services.ewsconsumer.schema.v4.creditsearch.creditsearchresponse.CreditSearchResponse;
import com.equifax.uk.services.ewsconsumer.schema.v4.commonrequest.CreditSearchConfig;
import com.equifax.uk.services.ewsconsumer.schema.v4.commonrequest.MatchCriteria;
import com.equifax.uk.services.ewsconsumer.schema.v4.commonrequest.MatchCriteriaStatus;
import com.equifax.uk.services.ewsconsumer.schema.v4.commonrequest.RequestPerson;
import com.equifax.uk.services.ewsconsumer.schema.v4.commonrequest.RequestSoleSearch;
import com.equifax.uk.services.ewsconsumer.schema.v4.creditsearch.creditsearchrequest.CreditSearchRequest;
import com.equifax.uk.services.ewsconsumer.schema.v4.datarequests.AssociateRequest;
import com.equifax.uk.services.ewsconsumer.schema.v4.datarequests.RequestedData;
import com.equifax.uk.services.ewssecurity.schema.v4.logon.logonrequest.LogonRequest;
import com.equifax.uk.services.ewssecurity.schema.v4.logon.logonresponse.LogonResponse;
import com.equifax.uk.services.schema.v4.common.address.MatchedResidentialStructuredAddress;
import com.equifax.uk.services.schema.v4.common.address.StructuredAddress;
```

```
import com.equifax.uk.services.schema.v4.common.bank.BankAccount;
import com.equifax.uk.services.schema.v4.common.person.BankingInfo;
import com.equifax.uk.services.schema.v4.common.person.ContactInfo;
import com.equifax.uk.services.schema.v4.common.person.Employer;
import com.equifax.uk.services.schema.v4.common.person.EmploymentInfo;
import com.equifax.uk.services.schema.v4.common.person.Name;
import com.equifax.uk.services.schema.v4.common.person.ResidenceInstance;
import com.equifax.uk.services.schema.v4.common.person.TelephoneNumber;
import com.equifax.uk.services.schema.v4.common.person.TelephoneNumberUsageType;
import com.equifax.uk.services.schema.v4.EWSConsumerService;
import com.equifax.uk.services.schema.v4.EWSConsumerService_Service;
import com.equifax.uk.services.schema.v4.EWSSecurityService;
import com.equifax.uk.services.schema.v4.EWSSecurityService_Service;
import com.equifax.uk.services.schema.v4.CreditSearchFault;
import java.util.logging.*;

public class ConsumerCreditSearchRequestTest {

        public static void main(String[] args) throws Exception {
                try {

                        URL url = new URL("https://api.uat.equifax.co.uk/efx-uk-xmlii-consumer/v1");
                        QName qName = new QName("http://services.uk.equifax.com/schema/v4", "EWSConsumerService");
                        EWSConsumerService_Service serviceProxy = new EWSConsumerService_Service(url,qName);

            EWSConsumerService consumerService = serviceProxy.getEWSConsumerServiceSoap11();

                        String accessToken = getAccessToken("client_id", "client_secret");

                                                Map<String,          Object>          requestContext          =
((BindingProvider)consumerService).getRequestContext();

                        Map<String, List<String>> headers = new HashMap<String, List<String>>();
                        headers.put("Authorization", Collections.singletonList("Bearer "+accessToken));
                        requestContext.put(MessageContext.HTTP_REQUEST_HEADERS, headers);

            CreditSearchRequest sampleCreditSearchRequest = createSampleRequest();

            CreditSearchResponse csr = consumerService.creditSearch(sampleCreditSearchRequest);
                }catch(CreditSearchFault ex) {
                        Logger.getLogger(ConsumerCreditSearchRequestTest.class.getName()).log(Level.FINE,  "Fault
returned by consumer service.");

                 Logger.getLogger(ConsumerCreditSearchRequestTest.class.getName()).log(Level.FINE, "Code:      " +
ex.getFaultInfo().getCode());

                  Logger.getLogger(ConsumerCreditSearchRequestTest.class.getName()).log(Level.FINE, "Message:  " +
ex.getFaultInfo().getMessage());

                }
        }

        private static String getAccessToken(String clientId, String clientSecret) throws Exception {
                URL url = new URL("https://api.uat.equifax.co.uk/efx-uk-xmlii-security/v1");
                QName qName = new QName("http://services.uk.equifax.com/schema/v4", "EWSSecurityService");
                LogonRequest logonRequest = new LogonRequest();
                logonRequest.setClientId(clientId);
                logonRequest.setClientSecret(clientSecret);
                EWSSecurityService_Service serviceProxy = new EWSSecurityService_Service(url, qName);
                EWSSecurityService securityService = serviceProxy.getEWSSecurityServiceSoap11();
                LogonResponse logonResponse = securityService.logon(logonRequest);
                return logonResponse.getAccesstoken();
        }

    private static CreditSearchRequest createSampleRequest() {
        CreditSearchRequest sampleCreditSearchRequest = new CreditSearchRequest();

        // request details
        sampleCreditSearchRequest.setClientRef("XML2_4_0_CM_SEARCH_TYPE_001");

        // Create the request sub-type
        RequestSoleSearch sampleRequestSoleSearch = new RequestSoleSearch();

        // Requested Data
        RequestedData sampleRequestedData = new RequestedData();

        // Data Types required
        AssociateRequest rd_as = new AssociateRequest();
        rd_as.setMaxNumber(0);
        sampleRequestedData.setAssociateRequest(rd_as);

        // Match Criteria
        MatchCriteria sampleMatchCriteria = new MatchCriteria();
        sampleMatchCriteria.setAssociate(MatchCriteriaStatus.REQUIRED);
        sampleMatchCriteria.setAttributable(MatchCriteriaStatus.NOT_REQUIRED);
        sampleMatchCriteria.setFamily(MatchCriteriaStatus.NOT_REQUIRED);
        sampleMatchCriteria.setPotentialAssociate(MatchCriteriaStatus.NOT_REQUIRED);
        sampleMatchCriteria.setSubject(MatchCriteriaStatus.REQUIRED);

        // Primary person
        RequestPerson sampleRequestPerson = new RequestPerson();

        BankingInfo sampleRequestPerson_BankInfo = new BankingInfo();

        BankAccount sampleRequestPerson_BankInfo_Account = new BankAccount();
        sampleRequestPerson_BankInfo_Account.setAccountNumber("12345678");
        sampleRequestPerson_BankInfo_Account.setSortCode("10-20-30");
        sampleRequestPerson_BankInfo.setAccount(sampleRequestPerson_BankInfo_Account);
        sampleRequestPerson.setBankingInfo(sampleRequestPerson_BankInfo);
        sampleRequestPerson_BankInfo.setTimeAtBank(null);
```

```
        // Contact details
        ContactInfo sampleRequestPerson_ContactInfo = new ContactInfo();

        TelephoneNumber sampleRequestPerson_ContactInfo_Telephone = new TelephoneNumber();
        sampleRequestPerson_ContactInfo_Telephone.setDirectoryListed(Boolean.TRUE);
        sampleRequestPerson_ContactInfo_Telephone.setNumber("300105");
        sampleRequestPerson_ContactInfo_Telephone.setPrefix("01324");
        sampleRequestPerson_ContactInfo_Telephone.setUsageType(TelephoneNumberUsageType.PERSONAL);
        sampleRequestPerson_ContactInfo.getTelephoneNumber().add(sampleRequestPerson_ContactInfo_Telephone);
        sampleRequestPerson.setContactInfo(sampleRequestPerson_ContactInfo);

        // Employment information
        EmploymentInfo sampleRequestPerson_Employment = new EmploymentInfo();
        sampleRequestPerson_Employment.setAnnualSalary(12000000);

        Employer sampleRequestPerson_Employment_Employer = new Employer();
        sampleRequestPerson_Employment_Employer.setName("COMERCIAL UNION INSURANCE");

        TelephoneNumber sampleRequestPerson_Employment_Employer_Telephone = new TelephoneNumber();
        sampleRequestPerson_Employment_Employer_Telephone.setNumber("300105");
        sampleRequestPerson_Employment_Employer_Telephone.setPrefix("01342");

sampleRequestPerson_Employment_Employer.setTelephoneNumber(sampleRequestPerson_Employment_Employer_Telephone);

        StructuredAddress sampleRequestPerson_Employment_Employer_Address = new StructuredAddress();
        sampleRequestPerson_Employment_Employer_Address.setDistrict("IMBERHONE");
        sampleRequestPerson_Employment_Employer_Address.setNumber("52");
        sampleRequestPerson_Employment_Employer_Address.setPostcode("RH191JS");
        sampleRequestPerson_Employment_Employer_Address.setPostTown("EAST GRINSTEAD");
        sampleRequestPerson_Employment_Employer_Address.setStreet1("CAMBELL CRESCENT");
        sampleRequestPerson_Employment_Employer.setAddress(sampleRequestPerson_Employment_Employer_Address);
        sampleRequestPerson_Employment.setEmployer(sampleRequestPerson_Employment_Employer);
        sampleRequestPerson.setCurrentEmployment(sampleRequestPerson_Employment);

        // Name of person to search
        Name sampleRequestPerson_Name = new Name();
        sampleRequestPerson_Name.setForename("DAVID");
        sampleRequestPerson_Name.setSurname("JONES");
        sampleRequestPerson.setName(sampleRequestPerson_Name);

        // Address of person
        ResidenceInstance sampleRequestPerson_ResidenceInstance = new ResidenceInstance();

                    MatchedResidentialStructuredAddress   sampleRequestPerson_ResidenceInstance_Address   =   new
MatchedResidentialStructuredAddress();
        sampleRequestPerson_ResidenceInstance_Address.setAddressID("28030032186");
        sampleRequestPerson_ResidenceInstance.setMatchedAddress(sampleRequestPerson_ResidenceInstance_Address);
        sampleRequestPerson.setCurrentAddress(sampleRequestPerson_ResidenceInstance);

        // Opt in/out selection
        CreditSearchConfig sampleRequestSoleSearch_Config = new CreditSearchConfig();
        sampleRequestSoleSearch_Config.setOptIn(true);

        // Add above sections to search request
        sampleRequestSoleSearch.setMatchCriteria(sampleMatchCriteria);
        sampleRequestSoleSearch.setRequestedData(sampleRequestedData);
        sampleRequestSoleSearch.setPrimary(sampleRequestPerson);
        sampleRequestSoleSearch.setCreditSearchConfig(sampleRequestSoleSearch_Config);

        // Add the solesearch to the request
        sampleCreditSearchRequest.setSoleSearch(sampleRequestSoleSearch);
        sampleCreditSearchRequest.setIsSubsequentRequest(Boolean.FALSE);

        return sampleCreditSearchRequest;
    }

}
```

4 Consumer Service

The request will be generated by the client, upon receipt of the request, it will be validated. If the validation fails, an error will be returned and the processing will finish. If all validation has been passed a bureau data search will be conducted on Equifax's database and a response in accordance with the details as specified within this guide, will be returned to the client system.

The WSDL of a web service cannot be viewed in the browser as per Equifax security policy. All the test and live environment static WSDL of Consumer Service are available as a .zip file at SDKs tabs. Select your preferred service WSDLs .zip file and download.

Only the latest version of the WSDLs and the URLs will be supported going forward, all clients who are using a version less than v4.0 will be required to migrate to v4.0 at the earliest opportunity.

4.1 Operations

There are a number of similar *operations* included within the Consumer Service, the key difference being the *type* of 'search' that is logged at Equifax during enquiry processing.

Successful enquiries (that is, those that have passed validation and uniquely address matched) will log a search on the Equifax database for each applicant at each of their matched address(es) containing the applicant name(s) and the client (service requester) details. DPA legislation mandates that the type of search logged must accurately reflect the purpose of the enquiry and so the correct Consumer Service operation must be used. The purpose of the search will also influence the level of data returned.

A search will NOT take place unless all supplied addresses (in the scope of the enquiry) match.

It is extremely important that the Consumer Service operation invoked accurately reflects the type of activity being performed and so new operations will be added as necessary. If you feel that the operations currently available do not reflect your activity, please speak with your Account Manager for advice.

The available operations are:

Operation	Search Type(s)
Credit Search	SR, SE
Credit Quotation	SQ
Customer Management	CM
Director Credit Search	DR
Application Fraud	FD
Anti Money Laundering	ML
Verify Identity Check	ID
Debt Collection	DC
Insurance Application	IN
Insurance Quotation	IQ
Personnel Vetting	PV
Tenant Verification	TV
Consumer Credit Search (Individual Credit File Access)	CZ
Fraud Investigation	FR
Id Check By Other Public Sector	N/A
Watchlist Check Full report request	N/A

Non-logging Operations

In addition to the above, there is an operation specifically for logging new Alias or Associate records with Equifax, where no search type is recorded:

- Alias Associate Notification

Each operation has defined web service request and response messages within the consumer service WSDL (Web Services Description Language) document which encapsulates the various XML Schema (XSD) documents that define the consumer service schema model.

4.2 Processing

Address Matching

If the request message is deemed to be valid, an attempt will be made to match the supplied addresses against the Equifax database.

To offer a high address matching rate the address matching service utilises many sophisticated techniques to cater for small spelling mistakes. To obtain the best address matching rate full address details should be provided. The input address format is not heavily restricted, however, the Information Commissioner's Office recommend that a full Post Office Address should be supplied.

For each input address, the result will be one of four possible situations:

- Unique match
- Pre matched
- Multiple match
- No match

Unique match means the supplied address matches to a single address on the Equifax database. Once an address has been matched, the required data retrieval activities will be performed for the enquiry.

Pre matched means the supplied address is already matched to a single address on the Equifax database.

A **multiple match** occurs when the supplied address matches to more than one address on the Equifax database. When a multiple match occurs no consumer data will be returned for the enquiry.

A **no match** occurs when the service has been unable to match the supplied address to any address on the Equifax database. If a supplied current address cannot be matched, no consumer data will be returned for the enquiry. The unmatched address should be amended or removed and the enquiry should be re-submitted.

The <continueOnFailedAddressMatch> flag allows the requester to indicate that the enquiry can continue, even if one or more of the **previous** address(es) fail to uniquely match. That is, the enquiry can continue to process using only the matching address(es).

Name Matching

The Data Protection Act 1998 imposed changes to the practices of the credit industry in the processing of personal data relating to third parties. Third party data in this context now means anyone other than the data *subject* .

Each piece of data returned by the Consumer Service will comply with guidelines agreed with the Information Commissioner's Office, which means that the data returned may vary depending upon the purpose of the search. Each data item returned will have a name match flag (except Telephone Data) to show the *level* of match to the name(s) supplied in the enquiry.

Each raw data record will be classified into one of the match levels (in order of strongest match) listed below:

Name match level	Description

Applicant	Data relates to the applicant.
Associate	Data relates to an associate of the applicant.
Potential Associate	Data relates to someone who may be an associate of the applicant.
Family	Data relates to a family member.
Attributable	Data found that could relate to one of the subjects of the search or another individual at the address.
Other	Any other data available to the enquiry. This data can not be requested and may be returned on Electoral Roll data only.

Where a raw data record can fall into more than one classification, it will be given the code highest in the table above.

The flags in the <matchCriteria> element allow the requester to define which name match level raw data should be returned in the response message.

> ⚠ These flag settings only relate to the return of raw data groups. That is, they will not affect which data is used in characteristic calculations or scoring.

Billing

Each successful request will result in a charge, and this will reflect the complexity of the request and the value of the information included. Your Equifax account manager can provide more precise details.

Alias and Associate logging

If the search is a joint search, or an associate name or an alias name have been supplied (or the Alias Associate Notification operation is being used) this information will be logged and may be used in subsequent searches by any Equifax client.

Locate records

If a current and previous address have been supplied and a *locate* address link does not already exist, then one will be logged and may be used in subsequent services by any Equifax client.

4.3 Data Groups

Equifax provide consumer data taken from a wide range of sources to enable its users to make informed decisions. This data can be returned as 'raw' data or 'coded' *characteristics* , which summarise the 'raw' data.

A check will be made for each data group requested to ensure that the requesting user has sufficient authority to request it. If the enquiry does not have authority to request a data group then this will be indicated in the response for the relevant data group with an <accessDenied> flag being returned. The search will not be terminated and details will be returned for each remaining valid data group.

The following data groups are available via the Consumer Service, in some cases these are subject to membership of the Closed User Groups, and are detailed in the quick reference section.

Non address-specific data groups

Data Group	Id
Alias	ALS
Associate	ASC
Attributable Names	ATT/ATV
Potential Associate	PAS
Link Names Notification	AAL
Scores	SCO
Watchlist Check Report	N/A
Watchlist Check Summary Report	N/A
Attribute Block	N/A
Watchlist Check Full Report	N/A

Address-specific data groups

Data Group	Supplied Address Id	Linked Address Id
Input Address	ADI	N/A
Output Address	ADO	ADV
Search	ASR	ASV
Court And Insolvency Information	CJR	CJV
CIFAS	CPR	CPV
Electoral Roll	ELR	ELV
Bank Check	FBC	N/A
Bureau Verification	FBV	N/A
Employer Check	FEC	N/A
Fraud Profile Summary	FPC	N/A
Fraud SIRA Check	FSC	N/A
Telephone Check Extended	FT1	N/A
Telephone Check	FTC	N/A
GAIN	GAN	GNV

ID Verification	IDC	IDV
Insight	INR	INV
InsightHistory	INY,IBR	INZ,IBV
Notice Of Correction Or Dispute	NCR	NCV
Property Valuation	PVR	PVV
Rolling Register	RRR	RRV
Telephone Data	TLR	N/A
AML Halo	VMC	N/A

⚠ Data groups FBC, FBV, FEC, FTC and FT1 can only be requested via the following operations: Credit Search

The watchlist check full report can only be requested in isolation using the Watchlist Check Full report request operation.

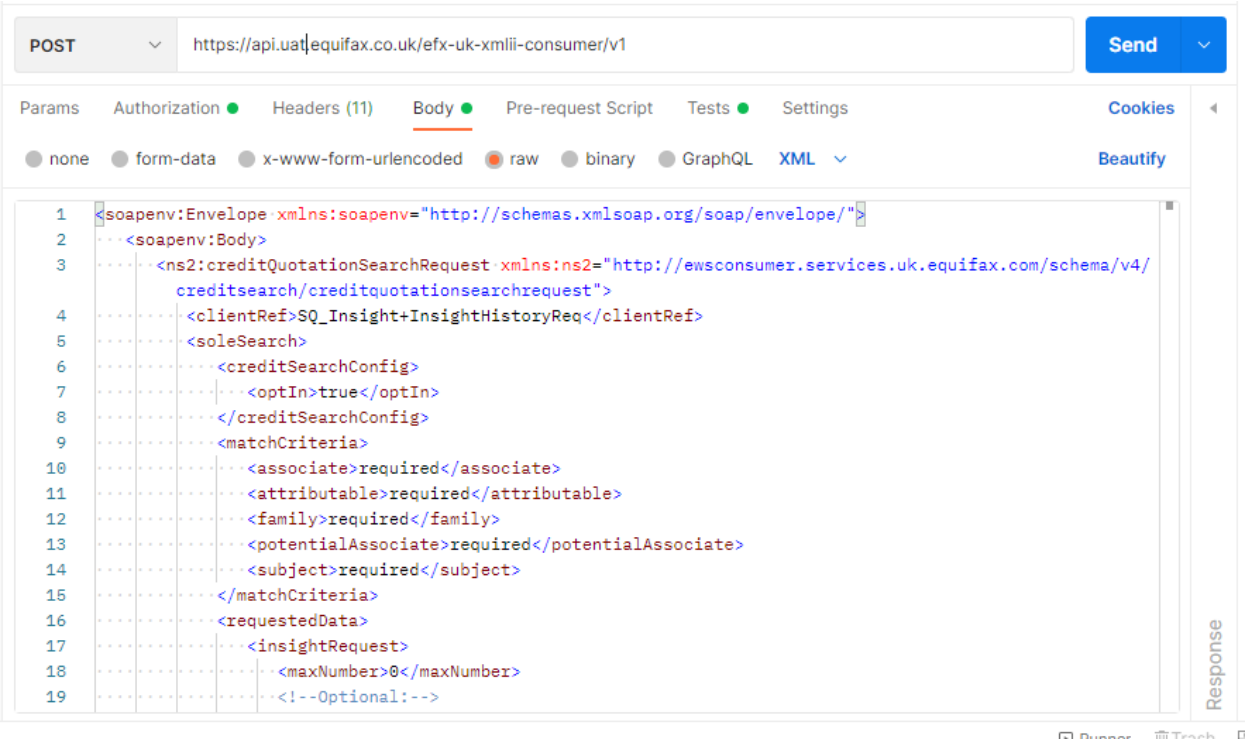
4.4 Request

Each of the operations share a number of common request elements. Full sample requests for each operation can be generated from the published WSDL document that describes the Consumer Service.

For example, sample Credit Search operation requests can be generated using the Postman web service testing tool, which also highlights optional elements as can be seen from the partial screenshot below:

⚠ Please be aware that the samples contained in the gateway have been generated via Postman, using the wsdls provided. Should you use an alternative application in conjunction with the wsdls, certain fields contained in the samples may not be required and are not generated by the wsdl, such as <header>.

Postman sample creditQuotationSearchRequest request message template



Request elements described

At the start of the request, there are two elements common to all types of enquiry.

Field	Required	Description
<clientRef>	Mandatory	Used to supply an enquiry reference identifier. The content of this element is the responsibility of the client using the service and the service will return this reference data as part of the response. This field is not stored and is returned without modification.
<storeIdentity>	Optional	Deprecated

Following these common request elements, the nature of the search (sole applicant application or joint, two applicant application) is indicated on the request via the choice of **<soleSearch>** or **<jointSearch>** *container* elements.

Both of these containers include the following four elements:

Field	Required	Description
<creditSearchConfig>	Optional	Indicates whether third party data (family or financial associates) should be used on an enquiry. Omitted on non-credit enquiries. The <optIn> flag will be checked to ensure it is consistent with other details requested (e.g. it will not be possible to opt out and request a household override score).
<matchCriteria>	Mandatory	Data matching criteria for a number of name match 'levels': applicant, associate, potential associate, attributable or family. Here, a restriction can be placed on the name match levels of raw data to be returned, except public data groups Electoral Roll and Rolling Register which by their nature allow access to **all** name match levels. Each name match level flag can be set to include/exclude or include/exclude depending upon whether a score has been calculated using data in that name match level.
<requestedData>	Mandatory	Request elements for each of the available data groups. A maximum number of records to be returned can be specified along with the address scope (supplied address data, linked address data or data from both).
<amlEnhancedWatchlistCheckProfile>	Optional	Indicates required usage of watchlist check product. <profile> will describe which of the pre-configured profiles is to be used when screening the customer. The valid values are GLOBAL, UKPLUS and UKSTD. Refer to the Watchlist Check Technical User Guide for more information.

There are some specific processing rules around the <matchCriteria> name match level flags:

1. At least one of the flags must be set to 'required' if a 'raw' data group has been requested.
2. At least one scoring product must be requested if any of the flags are set to 'onlyForScores'.
3. If Associate raw data is requested, then the associate flag must NOT be set to 'notRequired'.
4. If Potential Associate raw data is requested, then the potentialAssociate flag must NOT be set to 'notRequired'.
5. If Attributable Names raw data is requested, then the attributable flag must NOT be set to 'notRequired'.

⚠ When using the 'used in scoring' setting, if the conditions are not met, and therefore a score is not calculated, a default score value will be returned, and no raw data will be returned for that match level.

Both the <soleSearch> and <jointSearch> containers include the following applicant information within a **<primary>** containing element. The <jointSearch> element also contains a <primary> element for the primary applicant along with a **<relationship>** element indicating the nature of the relationship between the applicants, and a **<secondary>** element for the joint applicant (which has the same structure as the <primary> element).

The <relationship> element can indicate either a joint application, or a declared financial associate who is not party to this application. In both cases, data relating to the applicant and the applicants' associates can be included, so:

- If an **associate** is indicated, data will be retrieved for the first applicant and her associates, which will include the one provided on the enquiry.
- If a **joint** application is indicated, data will be retrieved for both applicants separately, which can result in the inclusion of data for both applicants' associates.

⛔ If the <relationship> element indicates an associate, the <optIn> field must be set to true.

Ordinarily, a joint application will result in a record being stored on the Equifax database to show that the two applicants have a financial association. In rare cases, the joint application may not be an indication of a financial association (e.g. the hiring of an appliance by a group of students) and in these cases this flag will be used to indicate the relationship is **transient** and that a financial association should NOT be stored on the Equifax database.

Applicant information

Field	Required	Description
<annualExpenditure>	Optional	Value in whole pounds for the year.
<bankingInfo>	Optional	Account and Bank details including the time the account has been held at the bank if known.
<contactInfo>	Optional	Up to two email addresses and three telephone numbers.
<currentEmployment>	Optional	Employment details including: salary, duration of employment, employer details, occupation and payment frequency.
<dob>	Optional	Date of birth.
<drivingLicenceNumber>	Optional	Driving licence number.
<maritalInfo>	Optional	Marital details including status and length of time married if applicable.
<mothersMaidenName>	Optional	Mother's maiden name.
<name>	Optional	Name details.
<nationalInsuranceNumber>	Optional	National insurance number.
<passportNumber>	Optional	Passport number.
<placeOfBirth>	Optional	Place of birth.
<previousEmployment>	Optional	Previous employment details including: salary, duration of employment, employer details, occupation and payment frequency.
<previousName>	Optional	Previous name details.
<sex>	Optional	Male or female.
<currentAddress>	Optional	Current address details including residential status and time at address. Address input can be supplied in one of three different formats.
<introducer>	Optional	Referrer details for credit application.
<loanDetail>	Optional	Loan details if appropriate - amount and purpose of loan.
<previousAddress>	Optional	Up to two previous addresses in a choice of formats as per current address.
<solicitor>	Optional	Solicitor details if appropriate - name and address.
<timeOnERCurrentAddress>	Optional	Amount of time on electoral roll at applicant's current address.
<nationality>	Optional	Nationality of the applicant.

<countryOfResidence>	Optional	Country of residence of the applicant, likely to be United Kingdom in many cases.

Sophisticated data filtering techniques will be used to maximise the data returned by a search; failure to provide full name details may adversely affect this processing.

⚠ The Information Commissioner's Office recommend that full name details, including title are supplied on all enquiries.

Address Structures

There are several ways to supply an address within each of the address *container* elements (**<currentAddress>**, **<address>**, **<previousAddress>**) in the service request.

Structured address

```
<address>
   <country>...</country>
   <county>...</county>
   <district>...</district>
   <name>...</name>
   <number>...</number>
   <poBox>...</poBox>
   <postcode>...</postcode>
   <postTown>...</postTown>
   <street1>...</street1>
   <street2>...</street2>
   <subBuilding>...</subBuilding>
   <residenceType>...</residenceType>
</address>
```

The above *address* structure is preferred as each (optional) component part of the address element is expressed explicitly, facilitating the best possible match to an Equifax-held address.

Free format address

```
<freeFormatAddress>
   <line>...</line>
   <line>...</line>
   …
   <residenceType>...</residenceType>
</freeFormatAddress>
```

If the input address is less structured, a *free format* address structure can be supplied with up to 6 <line> elements.

Matched address

```
<matchedAddress>
   …
   <addressID>
</matchedAddress>
```

Finally, a *matched* address structure can be supplied which is essentially the same as the <address> structure above, but with a mandatory **<addressID>** element which should contain an 11 character Equifax address key, obtained via an earlier enquiry.

Response message address structure

The matched address format is the structure used in response messages where a matched address is returned. However, the free format address format is the structure used to represent input addresses in the Input Address data group.

4.5 Response

The response message will vary depending on the operation invoked and the data, scores and characteristics requested.

⛔ If an error is encountered during processing, a **SOAP Fault** response will be generated. If the system is available, then the request will return a SOAP XML response message. If the system is unavailable, then one of a number of responses will be received: either **no response** whatsoever or an HTTP error response code depending on where the problem has been encountered.

Response elements described

Again, with the exception of the Change Client Secret operations, each of the operations share a number of common properties.

Field	Description
<clientRef>	The value supplied in the request is simply echoed back in the corresponding response.
<nonAddressSpecificData>	This section contains data groups not specific to either a supplied address or a linked address found during enquiry processing relating to a supplied address. Data groups included: Alias, Associate, Attributable, Link Names Notification, Potential Associate, Scores, Characteristics.

Following these common response elements, the nature of the search (sole applicant application or joint, two applicant application) is indicated on the response message via either a **<soleSearch>** or **<jointSearch>** *container* element.

Both the <soleSearch> and <jointSearch> containers include the response data for the first applicant in the enquiry within a **<primary>** containing element. The <jointSearch> element also contains a **<secondary>** element for the second applicant (which has the same structure as the <primary> element).

For each applicant, address-specific data is returned within either a **<suppliedAddressData>** block or a **<linkedAddressData>** block, depending on the address it relates to. These blocks are structured as follows:

Field	Description
<addressMatchStatus>	Multiple, Single, Pre or No match.
<addressSpecificData>	Contains requested data groups if found during enquiry processing.
<index>	Sequence number relating to the position of the supplied address or identified linked address.
<inputAddress> or <linkedAddress>	Free format address as supplied on input or matched address identifier relating to the linked address found.
<linkType>	Relationship of the linked address to one of the supplied addresses.
<noticeOfCorrectionPresent>	Indicates the presence of a notice of correction record at the address.
<matchedAddress> or <potentialMatchedAddress> or <unmatchedAddress>	One of three styles of returned address details depending on the match result of the supplied address.

4.6 Watchlist Check

Unlike other consumer service product offerings Equifax Watchlist Check supports a two stage enquiry process:

1.Summary Report – screens a prospective or existing customer against the Dow Jones Watchlist and returns a summary of the candidate matches.

2.Full Report – provides the full detail for a selected Dow Jones identity to enable a case review

The Summary Report can be requested using the existing Consumer Service options described above. The result of this transaction will be a report that provides an overview of the matches generated from the screening check. The relevant request items are **<amlEnhancedWatchlistCheckRequest>** and **<amlEnhancedWatchlistCheckSummary>**. These may be accessed from the data groups section of this document.

A Full Report can only be done on the back of a previous Summary Report where it has returned one or more match candidates, the request can be either immediately following the Summary Report or at some point after the event. The watchlist check full report can only be requested in isolation, and the request message is unique to this element of the product offering.

The Full Report is specific to an individual (or business entity) stored within the Dow Jones watchlist data, information about an individual or entity is compiled and consolidated to create under a unique profile. Usage of the watchlist check full report is described below.

4.6.1 Request

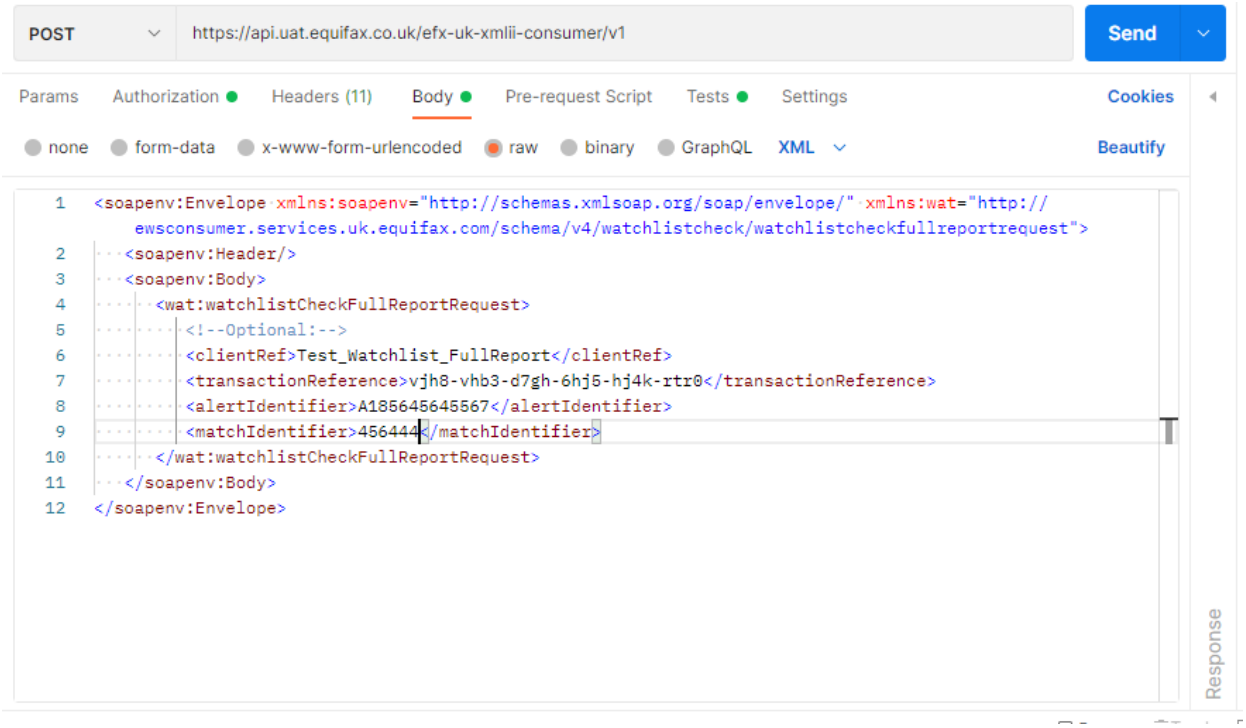
The watchlist check full report request will be formed from outputs extracted from the **<amlEnhancedWatchlistCheckRequest>** data group that may be requested when using the other operations that describe the consumer service.

A sample request for each this operation can be generated from the published WSDL document that describes the Consumer Service.

For example, sample Watchlist Check Full Report Request operation requests can be generated using the Postman web service testing tool, which also highlights optional elements as can be seen from the partial screenshot below:

> ⚠ Please be aware that the samples contained in the gateway have been generated via Postman, using the wsdls provided. Should you use an alternative application in conjunction with the wsdls, certain fields contained in the samples may not be required and are not generated by the wsdl, such as <header>.

Postman sample watchlistCheckFullReportRequest request message template:



Request elements described

The request elements are:

Field	Required	Description
<transactionReference>	Mandatory	A unique reference number for the transaction as a whole. All alerts and matches generated for a single Summary Report will return the same Transaction Reference.
<alertIdentifier >	Mandatory	A unique reference number for an alert. An alert can potentially generate multiple matches, e.g. where a check such as 'Global Sanctions (Individual Name & DOB & RSCY)' produces matches to more than one Dow Jones identity – in this scenario all matches would share the same Alert Identifier.
<matchIdentifier>	Mandatory	A reference number for a match. The combination of Transaction Reference, Alert Identifier and Match Identifier will always be unique.

For all request elements the contents should be as generated as part of a previously submitted enquiry via the Consumer service.

4.6.2 Response

The response message is described within the data groups section of this document (Watchlist Check Full Report).

If an error is encountered during processing, a **SOAP Fault** response will be generated. If the system is available, then the request will return a SOAP XML response message. If the system is unavailable, then one of a number of responses will be received: either **no response** whatsoever or an HTTP error response code depending on where the problem has been encountered.

5 Address Matching Service

Credit reference data (bureau data) is mainly related to an individual's current and previous addresses. Equifax bureau data is managed using a proprietary key value that uniquely identifies each known address (PTC-Abs: post town code, absolute house ID). Address matching is an important stage in the credit search process as it facilitates the resolution of real world address details to PTC-Abs codes. This ensures that any bureau data extracted is relevant only to the individual concerned. While other address verification methods are available from the numerous PAF licensees, they cannot provide the required PTC-Abs code.

It should be noted, the service described within this section relates to the provision and matching of residential address data only and cannot be used to resolve the addresses of commercial entities. The Address Matching Service is designed to be used in an interactive manner and if possible the results returned to the end-user to allow the correct address to be identified. The match status indicates how successful the address matching was and should be used to determine how the results are presented back. This is important where both multiple matches and help data are encountered.

It is therefore important if the address matching service is being utilised that your own system is designed to allow this level of interactivity within these situations. Without this interactivity, the ability to resolve an address to a single match will be limited and whilst this may not prevent any application process from continuing it will necessitate additional effort on the part of the operator.

The WSDL of a web service cannot be viewed in the browser as per Equifax security policy. All the test and live environment static WSDL of Address Service are available as a .zip file at SDKs tabs. Select your preferred service WSDLs .zip file and download.

Only the latest version of the WSDLs and the URLs will be supported going forward, all clients who are using a version less than v4.0 will be required to migrate to v4.0 at the earliest opportunity.

5.1 Address Match

Address Match

5.2 Address List

Address List

6 Security Service

The request will be generated by the client, upon receipt of the request, it will be validated. If the validation fails, an error will be returned and the processing will finish. If all validation has been passed, security operations will be conducted on Equifax's database and a response in accordance with the details as specified within this document will be returned to the client system.

The WSDL of a web service cannot be viewed in the browser as per Equifax security policy. All the test and live environment static WSDL of Security Service are available as a .zip file at SDKs tabs. Select your preferred service WSDLs .zip file and download.

Only the latest version of the WSDLs and the URLs will be supported going forward, all clients who are using a version less than v4.0 will be required to migrate to v4.0 at the earliest opportunity.

6.1 Logon

Logon requests a logon for the supplied client id and client secret. The client is logged on to the Equifax application architecture and a session is created. An access token that identifies this session is returned in the response. The access token must be supplied in all future requests made for this session. Once authenticated, for optimal performance we highly recommend using the same token for all subsequent consumer and address service calls for the tokens lifespan. Currently a token is valid for 5 hours (300 minutes) and we recommend a new one is requested just prior to expiry for continuity of service. If the supplied logon details are incorrect/expired then an error response will be returned.

6.2 Change Client Secret

The Change Client Secret operation is used to change the client secret for a client id.

This is particularly useful where an account has been reset in response to a customer support issue being raised as this will cause the client secret to be pre-expired and mandates that the client secret is changed before first use.

The client secret associated with the client id will be set to expire in accordance with the standard Equifax security policies. This secret generated will expire after 30 days, this will therefore require the client to employ the Change Client Secret operation, periodically, to ensure that the account does not become locked and therefore unusable.

7 Fault And Error Handling

> ⚠ Please be aware that the samples contained in the gateway have been generated via Postman, using the wsdls provided. Should you use an alternative application in conjunction with the wsdls, certain fields contained in the samples may not be required and are not generated by the wsdl, such as <header>.

An operation-specific SOAP fault (each declared in an appropriate XML Schema document associated with each web service) is reported by the services with a customised **detail** section indicating a code and message relating to the error condition encountered.

These faults are exposed as exceptions in the client API (EWSConsumerException for the Consumer Service and EWSAddressException for the Address Matching Service).

Example creditSearchFault returned for the creditSearch operation where an incorrect access token is supplied on the request

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
    <soapenv:Header/>
    <soapenv:Body>
      <soapenv:Fault>
         <faultcode>soapenv:Server</faultcode>
         <faultstring xml:lang="en">Invalid Token error</faultstring>
         <detail>
            <ns2:creditSearchFault xmlns:ns2="http://ewsconsumer.services.uk.equifax.com/schema/v4">
               <code>EWSC0005</code>
               <message>Invalid Access Token</message>
            </ns2:creditSearchFault>
         </detail>
      </soapenv:Fault>
   </soapenv:Body>
</soapenv:Envelope>
```

Example logonFault returned for the logon operation where an incorrect client id or client secret is supplied on the request

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
    <soapenv:Header/>
    <soapenv:Body>
      <soapenv:Fault>
         <faultcode>soapenv:Server</faultcode>
         <faultstring xml:lang="en">Request processing error</faultstring>
         <detail>
            <ns2:logonFault xmlns:ns2="http://ewssecurity.services.uk.equifax.com/schema/v4">
               <code>EWSS0005</code>
               <message>Invalid client id or client secret</message>
            </ns2:logonFault>
         </detail>
      </soapenv:Fault>
```

```
   </soapenv:Body>
</soapenv:Envelope>
```

Example changeclientsecretFault returned for the changeclientsecret operation where an incorrect client id or client secret is supplied on the request

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
   <soapenv:Header/>
   <soapenv:Body>
      <soapenv:Fault>
         <faultcode>soapenv:Server</faultcode>
         <faultstring xml:lang="en">Request processing error</faultstring>
         <detail>
            <ns2:changeclientsecretFault xmlns:ns2="http://ewssecurity.services.uk.equifax.com/schema/v4">
               <code>EWSS0005</code>
               <message>Invalid client id or client secret</message>
            </ns2:changeclientsecretFault>
         </detail>
      </soapenv:Fault>
   </soapenv:Body>
</soapenv:Envelope>
```

Example changeclientsecretFault returned for the changeclientsecret operation where an incorrect access token is supplied on the request

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
   <soapenv:Header/>
   <soapenv:Body>
      <soapenv:Fault>
         <faultcode>soapenv:Server</faultcode>
         <faultstring xml:lang="en">Invalid Token error</faultstring>
         <detail>
            <ns2:changeclientsecretFault xmlns:ns2="http://ewssecurity.services.uk.equifax.com/schema/v4">
               <code>EWSS0005</code>
               <message>Invalid access token</message>
            </ns2:changeclientsecretFault>
         </detail>
      </soapenv:Fault>
   </soapenv:Body>
</soapenv:Envelope>
```

Example logonFault returned for the logon operation when non-recoverable error occured during runtime

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
    <soapenv:Header/>
    <soapenv:Body>
      <soapenv:Fault>
         <faultcode>soapenv:Server</faultcode>
         <faultstring xml:lang="en">Internal web service error</faultstring>
         <detail>
            <ns2:logonFault xmlns:ns2="http://ewssecurity.services.uk.equifax.com/schema/v4">
               <code>EWSS9999</code>
               <message>Web service cannot process request</message>
            </ns2:logonFault>
         </detail>
      </soapenv:Fault>
   </soapenv:Body>
</soapenv:Envelope>
```

7.1 Error Codes

Consumer Service fault codes

Fault string	Fault detail code	Custom fault message	Notes
Request processing error	**EWSC0001**	*Varies - message generated by a service dependency*	This fault is returned when a service dependency encounters a problem and is not always recoverable by the client.
Request processing error	**EWSC0002**	Service initialisation error	This fault is returned when the service being requested fails to initialise and is not usually recoverable by the client.
Request message error	**EWSC0003**	*Varies - several possible error messages relating to the request content*	This fault indicates that the Consumer service request is incorrect or incomplete. The client should review request message content prior to submission.
Validation error	**EWSC0004**	*Varies - request or response message validation error depending on problem encountered*	This fault is returned when validation of the incoming request message or outgoing response message fails for some reason. The client should validate request messages against the service XML Schema documents prior to submission.

Access authentication error	**EWSC0005**	Invalid Access Token	This fault is returned in several circumstances relating to access token failure: expired access token, incorrect access token. Some of these conditions are recoverable by the client who should ensure that the correct access token are supplied on the request and that accounts are managed appropriately.
Internal web service error	**EWSC9999**	Web service cannot process request	This fault indicates a runtime or operational error that is not recoverable by the client.

Watchlist Check Full Report Service fault codes

Fault string	Fault detail code	Custom fault message	Notes
Request processing error	**EWSC0001**	*Varies - message generated by a service dependency*	This fault is returned when a service dependency encounters a problem and is not always recoverable by the client.
Request processing error	**EWSC0002**	Service initialisation error	This fault is returned when the service being requested fails to initialise and is not usually recoverable by the client.
Request message error	**EWSC0003**	*Varies - several possible error messages relating to the request content*	This fault indicates that the Consumer service request is incorrect or incomplete. The client should review request message content prior to submission.
Validation error	**EWSC0004**	*Varies - request or response message validation error depending on problem encountered*	This fault is returned when validation of the incoming request message or outgoing response message fails for some reason. The client should validate request messages against the service XML Schema documents prior to submission.
Access authentication error	**EWSC0005**	Invalid Access Token	This fault is returned in several circumstances relating to access token failure: expired access token, incorrect access token. Some of these conditions are recoverable by the client who should ensure that the correct access token are supplied on the request and that accounts are managed appropriately.
Watchlist Check enquiry lifecycle error	**EWSC0006**	Varies - request or response message validation error depending on problem encountered	This fault is returned when the information provided does not relate to a previously generated Watchlist Check summary report, generated via the Consumer service.
Internal web service error	**EWSC9999**	Web service cannot process request	This fault indicates a runtime or operational error that is not recoverable by the client.

Address Matching Service fault codes

Fault string	Fault detail code	Custom fault message	Notes
Request processing error	**EWSA0001**	*Varies - message generated by a service dependency*	This fault is returned when a service dependency encounters a problem and is not always recoverable by the client.
Request processing error	**EWSA0002**	Service initialisation error	This fault is returned when the service being requested fails to initialise and is not usually recoverable by the client.
Request message error	**EWSA0003**	*Varies - several possible error messages relating to the request content*	This fault indicates that the Consumer service request is incorrect or incomplete. The client should review request message content prior to submission.
Validation error	**EWSA0004**	*Varies - request or response message validation error depending on problem encountered*	This fault is returned when validation of the incoming request message or outgoing response message fails for some reason. The client should validate request messages against the service XML Schema documents prior to submission.
Access authentication error	**EWSC0005**	Invalid Access Token	This fault is returned in several circumstances relating to access token failure: expired access token, incorrect access token . Some of these conditions are recoverable by the client who should ensure that the correct access token are supplied on the request and that accounts are managed appropriately.
Internal web service error	**EWSA9999**	Web service cannot process request	This fault indicates a runtime or operational error that is not recoverable by the client.

Security Service fault codes

Fault string	Fault detail code	Custom fault message	Notes
User authentication error	**EWSS0005**	Invalid client id or client secret	This fault is returned in several circumstances relating to authentication failure: expired client secret, incorrect client secret, unknown client id, disabled client id. Some of these conditions are recoverable by the client who should ensure that the correct client id and client secret are supplied on the request and that accounts are managed appropriately.
Access authentication error	**EWSS0005**	Invalid Access token	This fault is returned in several circumstances relating to access token failure: expired access token, incorrect access token. Some of these conditions are recoverable by the client who should ensure that the correct access token are supplied on the request and that accounts are managed appropriately.
Internal web service error	**EWSS9999**	Web service cannot process request	This fault indicates a runtime or operational error that is not recoverable by the client.

8. Data Groups And Data Dictionary

The following data groups are available via the Consumer Service, in some cases these are subject to membership of the Closed User Groups.

Data Groups

Data Group

Alias
AML Halo
Associate
Attributable Names
Attribute Block
Bank Check
Bureau Verification
CIFAS
Court And Insolvency Information
Electoral Roll
Employer Check
Fraud Profile Summary
Fraud SIRA Check
GAIN
ID Verification
Input Address
Insight
InsightHistory
Link Names Notification
Notice Of Correction Or Dispute
Output Address
Potential Associate
Property Valuation
Rolling Register
Scores
Search
Telephone Check Extended
Telephone Check
Telephone Data
Watchlist Check Full Report
Watchlist Check Report
Watchlist Check Summary Report

Data Dictionary

Data Dictionary
CCJ Data Types
CIFAS Data Matching Rules
CIFAS Case Types
CIFAS Facilities
CIFAS Filing Reasons
CIFAS Fraud Address Types
CIFAS Subject Role Qualifiers
CIFAS Subject Roles
Company Class Types
Company Types
Insight Payment Statuses
Name Match Levels
SIC Codes