Oracle® Retail Brand Compliance Management Cloud Service

Implementation Guide

Release 1.10

E64991-04

April 2020

ORACLE®

Oracle Retail Brand Compliance Management Cloud Service Implementation Guide, Release 1.10

E64991-04

Primary Author: Bernadette Goodman

Contributing Author: Simon Tucker, Aidan Ratcliffe

Applications. For purposes of this section, "alteration" refers to all alterations, translations, upgrades, enhancements, customizations or modifications of all or any portion of the VAR Applications including all reconfigurations, reassembly or reverse assembly, re-engineering or reverse engineering and recompilations or reverse compilations of the VAR Applications or any derivatives of the VAR Applications. You acknowledge that it shall be a breach of the agreement to utilize the relationship, and/or confidential information of the VAR Applications for purposes of competitive discovery.

The VAR Applications contain trade secrets of Oracle and Oracle's licensors and Customer shall not attempt, cause, or permit the alteration, decompilation, reverse engineering, disassembly or other reduction of the VAR Applications to a human perceivable form. Oracle reserves the right to replace, with functional equivalent software, any of the VAR Applications in future releases of the applicable program.

Contents

4 RESTful APIs

A Appendix: Testing and Developing Clients for APIs

B Appendix: Using the Data Privacy API

Send Us Your Comments

Oracle Retail Brand Compliance Management Cloud Service Implementation Guide, Release 1.10

Oracle welcomes customers' comments and suggestions on the quality and usefulness of this document.

Your feedback is important, and helps us to best meet your needs as a user of our products. For example:

- Are the implementation steps correct and complete?
- Did you understand the context of the procedures?
- Did you find any errors in the information?
- Does the structure of the information help you with your tasks?
- Do you need different information or graphics? If so, where, and in what format?
- Are the examples correct? Do you need more examples?

If you find any errors or have any other suggestions for improvement, then please tell us your name, the name of the company who has licensed our products, the title and part number of the documentation and the chapter, section, and page number (if available).

> **Note:** Before sending us your comments, you might like to check that you have the latest version of the document and if any concerns are already addressed. To do this, access the Online Documentation available on the Oracle Technology Network web site. It contains the most current Documentation Library plus all documents revised or released recently.

Send your comments to us using the electronic mail address: retail-doc_us@oracle.com

Please give your name, address, electronic mail address, and telephone number (optional).

If you need assistance with Oracle software, then please contact your support representative or Oracle Support Services.

If you require training or instruction in using Oracle software, then please contact your Oracle local office and inquire about our Oracle University offerings. A list of Oracle offices is available on our web site at http://www.oracle.com.

Preface

This Implementation Guide describes the Application Programming Interfaces (APIs) available for this Oracle Retail Brand Compliance Management Cloud Service release.

Audience

This Implementation Guide is intended for the users of the Oracle Retail Brand Compliance Management Cloud Service application integration and implementation staff, and other parties intending to integrate with Brand Compliance Management Cloud Service.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Documents

For more information, see the following documents in the Oracle Retail Brand Compliance Management Cloud Service Release 1.10 documentation set:

- *Oracle Retail Brand Compliance Management Cloud Service Administration Guide*
- *Oracle Retail Brand Compliance Management Cloud Service Release Notes*
- *Oracle Retail Brand Compliance Management Cloud Service User Guide*

For information on the Oracle Retail Brand Compliance Management Cloud Service applications, see the following documents:

- *Oracle Retail Brand Compliance Management Cloud Service myProduct User Guide*
- *Oracle Retail Brand Compliance Management Cloud Service myProject User Guide*
- *Oracle Retail Brand Compliance Management Cloud Service myReports User Guide*
- *Oracle Retail Brand Compliance Management Cloud Service mySupplier User Guide*

Customer Support

To contact Oracle Customer Support, access My Oracle Support at the following URL:

https://support.oracle.com

When contacting Customer Support, please provide the following:

- Product version and program/module name
- Functional and technical description of the problem (include business impact)
- Detailed step-by-step instructions to re-create
- Exact error message received
- Screen shots of each step you take

Review Patch Documentation

When you install the application for the first time, you install either a base release (for example, 1.10) or a later patch release (for example, 1.10.1). If you are installing the base release or additional patches, read the documentation for all releases that have occurred since the base release before you begin installation. Documentation for patch releases can contain critical information related to the base release, as well as information about code changes since the base release.

Improved Process for Oracle Retail Documentation Corrections

To more quickly address critical corrections to Oracle Retail documentation content, Oracle Retail documentation may be republished whenever a critical correction is needed. For critical corrections, the republication of an Oracle Retail document may at times **not** be attached to a numbered software release; instead, the Oracle Retail document will simply be replaced on the Oracle Technology Network Web site, or, in the case of Data Models, to the applicable My Oracle Support Documentation container where they reside.

This process will prevent delays in making critical corrections available to customers. For the customer, it means that before you begin installation, you must verify that you have the most recent version of the Oracle Retail documentation set. Oracle Retail documentation is available on the Oracle Technology Network at the following URL:

http://www.oracle.com/technetwork/documentation/oracle-retail-100266.html

An updated version of the applicable Oracle Retail document is indicated by Oracle part number, as well as print date (month and year). An updated version uses the same part number, with a higher-numbered suffix. For example, part number E123456-02 is an updated version of an document with part number E123456-01.

If a more recent version of the document is available, that version supersedes all previous versions.

Oracle Retail Documentation on the Oracle Technology Network

Oracle Retail product documentation is available on the following web site:

http://www.oracle.com/technetwork/documentation/oracle-retail-100266.html

(Data Model documents are not available through Oracle Technology Network. You can obtain these documents through My Oracle Support.)

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1
Introduction

Note: The rebranding for the latest version of this documentation set is in development as part of post MICROS acquisition activities. References to former MICROS product names may exist throughout the existing documentation set.

Oracle Retail Brand Compliance Management Cloud Service is an integrated suite of applications designed to meet all aspects of sourcing, developing, and protecting retailer brands. The suite provides solutions for product development, compliance, quality, and traceability. It is designed specifically for retail, food service, and manufacturing businesses to develop and protect their brands, manage their suppliers, and ensure full end-to-end product lifecycle management.

Brand Compliance Management Cloud Service includes a number of exposed APIs that can be called by external systems. This document details all of the exposed APIs and the information required in order to access them.

Contents of this Guide

This implementation guide addresses the following topics:

- Chapter 2, "API Overview and Architecture": Overview of the Brand Compliance APIs and architecture used.
- Chapter 3, "SOAP APIs": Details of the available APIs that are based on the SOAP architecture.
- Chapter 4, "RESTful APIs": Details of the available APIs that are based on the RESTful architecture.
- Appendix A, "Appendix: Testing and Developing Clients for APIs": Guidance for the development and testing of clients to access the Brand Protection APIs.
- Appendix B, "Appendix: Using the Data Privacy API": Additional detail on using the Data Privacy API, including examples.

Key Features of Brand Compliance Management Cloud Service

The suite is composed of the following applications:

- myLibrary enables the issue, receipt, and acceptance of policies, guidelines, and key working documents.
- myProduct supports the development of products and production specifications.

- myProject supports the development of project briefs, plans, and workflow management.
- mySupplier enables the identification, selection, and approval of suppliers.

2

API Overview and Architecture

The APIs exposed by Brand Compliance are implemented in one of two formats, SOAP web services or RESTful web services. In both cases, requests and responses are normally in an XML format.

SOAP and RESTful APIs

Brand Compliance pushes APIs as web services in both SOAP and RESTful architectural styles. In general, the SOAP services are developed for specific requirements or are historical services. Unless there is a definite requirement for a SOAP implementation of a service, any new services or new versions of services are implemented using the RESTful architectural style.

Request and Response XML

In general, APIs exposed by Brand Compliance as web services use XML messages for both requests and responses. Exceptions to this are RESTful services where the use of an XML request is replaced by the use of URI parameters in a GET call.

The XML data schemas are written to allow for both forwards and backwards compatibility. All elements within the XML schemas are marked as optional. The element names are ordered so that new fields are added after existing elements and, within each sequence of elements, the last element is always an *any* element so that fields added in future versions are ignored by clients that are using an old version of the schema. For example:

```
<xs:complexType name="fpsMicroTestLocaleDataFullDTO">
    <xs:sequence>
        <xs:element form="qualified" minOccurs="0" name="description"
type="xs:string"/>
        <xs:element form="qualified" minOccurs="0" name="locale"
type="xs:string"/>
        <xs:element form="qualified" minOccurs="0" name="id" type="xs:long"/>
        <xs:element form="qualified" minOccurs="0" name="createdOn"
type="xs:string"/>
        <xs:element form="qualified" minOccurs="0" name="updatedOn"
type="xs:string"/>
        <xs:any maxOccurs="unbounded" minOccurs="0" namespace="##other"
processContents="lax"/>
    </xs:sequence>
  </xs:complexType>
```

The element names in the XML Schemas are defined by the core system and do not change in specific implementations. If an implementation has additional fields that

have been added to the system (the implementation is non-core), then those fields may be exposed after the core fields (before the xs:any element) and will incur a maintenance overhead in later versions to cope with any additional core fields added after the custom fields.

Data types in the XML Schemas are the standard datatypes for XSDs, WSDLs, and WADLs. Values should be entered as described on the following web site:

http://www.w3.org/TR/xmlschema-2/

The following table lists the commonly used types in Brand Compliance web service schemas.

Type	Example
xs:string	A normal string
xs:string	<![CDATA[A String with html data]]>
xs:long	1234
xs:int	1234
xs:short	1234
xs:boolean	true
xs:boolean	1
xs:date	2015-01-30
xs:dateTime	2015-01-30T23:59:59

Versioning

Due to the format of the XML schemas as described above, most revisions of the web services should be forwards and backwards compatible to allow clients of those web services to be upgraded and developed without binding the lifecycle of the client services to the upgrade cycle of Brand Compliance.

Where the necessary changes to a service cannot be made compatible with previous versions, a new endpoint will be created to expose the changed service. If possible, the previous version of the service will remain available at the original endpoint to enable existing and legacy clients to continue to work, but will be noted as deprecated in documentation to encourage developers of clients to the services to avoid using it.

Logging

All calls to APIs exposed as web services are logged within the application and can be viewed by logging into Brand Compliance as a Power User, navigating to myCompany > Admin> Notifications, and choosing the Web Service Log option. For each call, the name of the web service, date and time of the call, status (in progress/completed/failed), duration of the call (not available for In Progress calls), and a list of error messages if the call failed are shown.

The request and response XMLs are attached to each web service log entry. If a request does not contain XML, no attachment is created.

Accessing the APIs

This section describes how to access the APIs.

Deployment Model

The APIs, made available by Brand Compliance, are all web services implemented using either a SOAP or RESTful architecture.

The web services are exposed over HTTPS and require HTTP Basic authentication in order to access the service requested.

Authentication

In order to authenticate with Brand Compliance when accessing an API deployed as a web service, a user name and password must be supplied for HTTP basic authentication. Within Brand Compliance, the External System record defines the user names and passwords that have access to the web services.

To see the list of existing External System records, log in to Brand Compliance as a Power User, navigate to myCompany > Admin > Roles and Permissions, and select the External Systems option.

To create a new External System:

1. Select the Actions > New action to open up a tab for creating an External System.
2. Enter a Login ID (User Name) and select the Actions > Save option.
3. To set the password to be used by the External System, select the Actions > Change Password option.
4. Select the Actions > Save action. The new user name and password are active and available for use.

The passwords used for External Systems are validated against the same criteria as the passwords for interactive users of Brand Compliance. It is likely that a minimum length, mixed case, and digits will be required for the password, depending on the setup of the system.

Authorization

At present, all APIs are available to all authenticated requesters.

Listing Available APIs

In all installations of Brand Compliance, a full list of the deployed web services can be found at /services.

For example, for a system where the application is deployed to URL https://www.example.com/brandCompliance, you can retrieve a list of all services from https://www.example.com/brandCompliance/services.

Security Considerations

This section provides information on aspects of security you need to consider.

Password Management

Unlike interactive logins for users, the use of web services does not permit warnings to be included in the returned messages. Also, there is no guarantee that there is someone handling the call in order to interpret any warnings about password expiry. Therefore, Brand Compliance does not apply the password lifetime rules to External Systems.

Note: It is recommended that the passwords used for accessing APIs deployed as web services are changed regularly and that passwords are not reused or changed in such a way that the next value is predictable, such as, do not change from Password11 to Password12.

The passwords used to access the Brand Compliance APIs should be stored securely to prevent potential attackers from being able to masquerade as an authorized system.

Rich Text Data

In Brand Compliance, certain fields allow the display of HTML-formatted data. Such data is cleansed before display in Brand Compliance to ensure that any attempts at Cross-Site Scripting (XSS) attacks are mitigated, but this cleansing is not applied within the APIs. Therefore, it is advisable to cleanse any HTML-formatted data before it is displayed to users.

Excluded Fields

Many of the APIs published by Brand Compliance provide access to manage all of the records of a given type. With certain records, the fields accessible in the API are a subset of the fields within Brand Compliance. Any updates leave those fields unchanged. An example of this is the password-related fields on the User record, none of which are exposed in the APIs.

Secured Connections

As all of the APIs are exposed as web services over HTTPS, it is necessary for callers to communicate over SSL/TLS using the ciphers that the Brand Compliance servers accept. To ensure optimal security, it is advised that standard certificate checks are in place and not disabled. In the case of Apache CXF, the Client TLS Parameter disableCNCheck should be left as false.

3

SOAP APIs

The following SOAP APIs are available:

- User Details - Users
- Retrieve Product Specifications - Product Specifications
- Product Service - Product Records
- myArtwork Activities - Project Activities

User Details

This section describes the API used for user details.

userDetails

Description

This API is used in the myReporting Single Sign On integration with the TIBCO JasperSoft JasperReports Server to allow the JasperReports Server to retrieve the user information for the user that is logged into Brand Compliance. A oneTimePad is given to the JasperReports Server for use in calling this API. For each oneTimePad, a single call is permitted. Attempting to use the same oneTimePad multiple times results in failures.

Available from Version

1.0

Endpoint Address

/services/userDetails

Operation

getUser

Request Details

The request contains a single value, oneTimePad, which is the identifier passed to the JasperReports Server when the myReporting tab is opened.

Response Details

- If the oneTimePad value matches a value stored in the application, the details of the current logged in user are returned to the caller.

- If the oneTimePad value is not found, a fault is returned stating *Value not recognised*.
- If the user the oneTimePad refers to is not found, a fault is returned stating *userId cannot be null*.

Retrieve Product Specifications

This section describes the API for Product Specifications.

getProductSpecificationV1

Description

Retrieves a paged list of Product Specification records filtered according to the parameters defined in the request.

Available from Version

1.8

Endpoint Address

/services/getProductSpecificationV1

Operation

getProductSpecification

Request Details

Example request:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:v1="v1.getproductspecification.service.creations.micros.com">
   <soapenv:Header/>
   <soapenv:Body>
      <v1:getProductSpecification>
         <!--Optional:-->
         <request>
            <batchsize>?</batchsize>
            <offset>?</offset>
            <!--Zero or more repetitions:-->
            <specStatus>?</specStatus>
            <!--Zero or more repetitions:-->
            <specType>?</specType>
            <!--Zero or more repetitions:-->
            <countryWhereSold>?</countryWhereSold>
            <!--Zero or more repetitions:-->
            <productNumber>?</productNumber>
            <!--Zero or more repetitions:-->
            <language>?</language>
            <!--Zero or more repetitions:-->
            <sectionType>?</sectionType>
            <!--Zero or more repetitions:-->
            <productCoverage>?</productCoverage>
            <!--Zero or more repetitions:-->
            <supplier>?</supplier>
            <!--Zero or more repetitions:-->
            <site>?</site>
            <!--Optional:-->
```

```
            <fromDate>?</fromDate>
            <!--Optional:-->
            <toDate>?</toDate>
         </request>
      </v1:getProductSpecification>
   </soapenv:Body>
</soapenv:Envelope>
```

Request Parameters

Filter Type	Parameter Name	Mandatory/ Optional	Value Type	Example	Validation
Selection	offset	Mandatory	Numeric	60	>=0
Selection	batchSize	Mandatory	Numeric	30	>0 and <=100
Selection	specStatus	Optional	Multi Value String	RETAILER_ DRAFT	SUPPLIER_DRAFT RETAILER_DRAFT COLLABORATIVE_DRAFT GATE_STEP PART_PACK_COPY_SENT PACK_COPY_SENT PACK_COPY_READY READY_FOR_AUTHORISATION SUPPLIER_AUTHORISED ACTIVE OFF_RANGE DE_LISTED SUPERSEDED NOT_PROGRESSED APPROVE_FOR_LABELLING PRODUCE_DRAFT PRODUCE_PACK_COPY PRODUCE_APPROVED PRODUCE_ARCHIVED
Selection	specType	Optional	Multi Value String	FOOD	FOOD FNF CNF PRODUCE BWS
Selection	countryWhereSold	Optional	Multi Value String	FOOD_ COUNTRY_ WHERE_ SOLD	The Code for the required Country Where Sold.
Selection	productNumber	Optional	Multi Value String	12345	Filters on the Product Number column of the Product Coverage table.
Response	language	Optional	Multi Value String	en_GB	The Code for the required Language.

Filter Type	Parameter Name	Mandatory/ Optional	Value Type	Example	Validation
Response	sectionType	Optional	Multi Value String	RECIPE AND RAW MATERIALS	MAIN DETAILS RECIPE AND RAW MATERIALS NUTRITION ALLERGY AND DIETARY ADVICE PACKAGING FINISHED PRODUCT STANDARDS STORAGE OTHER LABELLING COPY CLAIMS SUBSTANTIATION PROCESS CONTROLS BATCH CODING PRODUCT APPROVAL REQUIREMENTS POST LAUNCH INFORMATION COMPONENTS FNF STORAGE PRODUCT REQUIREMENTS COUNTER TICKET PRODUCT CHARACTER COMP
Selection	productCoverage	Optional	Multi Value String	Chicken (500g)	Filters on the Product Name column of the Product Coverage table.
Selection	supplier	Optional	Multi Value String	A0001	Filters on the Primary Sites supplier code for Produce Specifications or the Supplier code for all other types of Specification.
Selection	site	Optional	Multi Value String	A0001-001	Filters on the code of the Sites in the Primary Sites table of the Product Specification.
Selection	fromDate	Optional	String	2013-06-10T09:00:00	Can be used in conjunction with toDate to form a date range or can be specified individually.
Selection	toDate	Optional	String	2013-06-10T09:00:00	Can be used in conjunction with fromDate to form a date range or can be specified individually.

The offset and batchSize parameters control the paging of the list. The offset defines the first record to retrieve (0-based) and the batchSize parameter defines how many records to include in the list.

Searching with either fromDate or toDate specified, as well as one or more specStatus, filters the Product Specification data on the Status Change History. For example:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:v1="v1.getproductspecification.service.creations.micros.com">
    <soapenv:Header/>
    <soapenv:Body>
```

```
        <v1:getProductSpecification>
            <request>
                <batchsize>50</batchsize>
                <offset>0</offset>
                <specStatus>ACTIVE</specStatus>
                <specStatus>OFF_RANGE</specStatus>
                <fromDate>2013-05-01T00:00:00</fromDate>
                <toDate>2013-05-31T24:00:00</toDate>
            </request>
        </v1:getProductSpecification>
    </soapenv:Body>
</soapenv:Envelope>
```

This example provides the first 50 Product Specifications that changed status to either Active or Off Range in May 2013.

Searching with either fromDate or toDate specified without specStatus filters on the last amended date of the Product Specification.

Response Details

For a successful response, XML is returned describing each Specification matching the supplied filters. The response XML is post-processed based on any Response filter types to exclude unwanted data. For example:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns0:getProductSpecificationResponse
 xmlns:ns2="http://www.micros.com/creations/core/domain/dto/v1p0/full"
 xmlns:ns3="http://www.micros.com/creations/core/domain/dto/v1p0/simple"
 xmlns:ns0="v1.getproductspecification.service.creations.micros.com">
         <return>
            <productsMatched>4</productsMatched>
            <ret>
               <ns2:isMultipack>false</ns2:isMultipack>
               <ns2:legislation>MODULE_TYPE_EU</ns2:legislation>
               <ns2:packCopyLanguage>
                  <ns3:code>en_GB</ns3:code>
```

Product Service

This section describes the APIs for Product Records.

createProduct

Description

A service to allow the creation of Product Records.

Available from Version

1.8

Endpoint Address

/services/productService

Operation

createProduct

Request Details

The request is an XML document. The following example shows the values that may be supplied:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:ext="external.service.creations.micros.com"
 xmlns:sch="http://www.micros.com/schema">
   <soapenv:Header/>
   <soapenv:Body>
      <ext:createProductRequest>
         <product>
            <sch:ProductNumber>?</sch:ProductNumber>
            <sch:ProductName>?</sch:ProductName>
            <sch:SupplierCode>?</sch:SupplierCode>
            <!--Optional:-->
            <sch:SupplierName>?</sch:SupplierName>
            <!--Optional:-->
            <sch:Status>?</sch:Status>
            <!--Zero or more repetitions:-->
            <sch:ProductBarcode>?</sch:ProductBarcode>
            <!--Zero or more repetitions:-->
            <sch:OuterBarcode>?</sch:OuterBarcode>
            <!--Optional:-->
            <sch:SpecificationType>?</sch:SpecificationType>
            <!--Optional:-->
            <sch:SpecificationTypeFormat>?</sch:SpecificationTypeFormat>
            <sch:ProductTechnologist>?</sch:ProductTechnologist>
            <!--Optional:-->
            <sch:OtherContacts>
               <!--Zero or more repetitions:-->
               <sch:Contact>
                  <!--1 or more repetitions:-->
                  <sch:Name>?</sch:Name>
                  <sch:Role>?</sch:Role>
               </sch:Contact>
            </sch:OtherContacts>
            <!--Optional:-->
            <sch:ProjectName>?</sch:ProjectName>
            <sch:Quantity>?</sch:Quantity>
            <!--Optional:-->
            <sch:Sites>
               <!--Zero or more repetitions:-->
               <sch:SiteCode>?</sch:SiteCode>
            </sch:Sites>
            <!--Optional:-->
            <sch:LeadBusinessCategory>?</sch:LeadBusinessCategory>
            <!--Optional:-->
            <sch:BusinessCategories>
               <!--Zero or more repetitions:-->
               <sch:Category>?</sch:Category>
            </sch:BusinessCategories>
            <!--Zero or more repetitions:-->
            <sch:CountryWhereSold>?</sch:CountryWhereSold>
            <!--Optional:-->
            <sch:UserDefinedFieldsData>
               <!--Optional:-->
               <sch:dateField001>?</sch:dateField001>
               <!--Optional:-->
               <sch:dateField002>?</sch:dateField002>
               <!--Optional:-->
```

```
<sch:dateField003>?</sch:dateField003>
<!--Optional:-->
<sch:dateField004>?</sch:dateField004>
<!--Optional:-->
<sch:dateField005>?</sch:dateField005>
<!--Optional:-->
<sch:dateField006>?</sch:dateField006>
<!--Optional:-->
<sch:dateField007>?</sch:dateField007>
<!--Optional:-->
<sch:dateField008>?</sch:dateField008>
<!--Optional:-->
<sch:dateField009>?</sch:dateField009>
<!--Optional:-->
<sch:dateField010>?</sch:dateField010>
<!--Optional:-->
<sch:dateField011>?</sch:dateField011>
<!--Optional:-->
<sch:dateField012>?</sch:dateField012>
<!--Optional:-->
<sch:dateField013>?</sch:dateField013>
<!--Optional:-->
<sch:dateField014>?</sch:dateField014>
<!--Optional:-->
<sch:dateField015>?</sch:dateField015>
<!--Optional:-->
<sch:dateField016>?</sch:dateField016>
<!--Optional:-->
<sch:dateField017>?</sch:dateField017>
<!--Optional:-->
<sch:dateField018>?</sch:dateField018>
<!--Optional:-->
<sch:dateField019>?</sch:dateField019>
<!--Optional:-->
<sch:dateField020>?</sch:dateField020>
<!--Optional:-->
<sch:longField001>?</sch:longField001>
<!--Optional:-->
<sch:longField002>?</sch:longField002>
<!--Optional:-->
<sch:longField003>?</sch:longField003>
<!--Optional:-->
<sch:longField004>?</sch:longField004>
<!--Optional:-->
<sch:longField005>?</sch:longField005>
<!--Optional:-->
<sch:longField006>?</sch:longField006>
<!--Optional:-->
<sch:longField007>?</sch:longField007>
<!--Optional:-->
<sch:longField008>?</sch:longField008>
<!--Optional:-->
<sch:longField009>?</sch:longField009>
<!--Optional:-->
<sch:longField010>?</sch:longField010>
<!--Optional:-->
<sch:longField011>?</sch:longField011>
<!--Optional:-->
<sch:longField012>?</sch:longField012>
<!--Optional:-->
```

```
<sch:longField013>?</sch:longField013>
<!--Optional:-->
<sch:longField014>?</sch:longField014>
<!--Optional:-->
<sch:longField015>?</sch:longField015>
<!--Optional:-->
<sch:longField016>?</sch:longField016>
<!--Optional:-->
<sch:longField017>?</sch:longField017>
<!--Optional:-->
<sch:longField018>?</sch:longField018>
<!--Optional:-->
<sch:longField019>?</sch:longField019>
<!--Optional:-->
<sch:longField020>?</sch:longField020>
<!--Optional:-->
<sch:numberField001>?</sch:numberField001>
<!--Optional:-->
<sch:numberField002>?</sch:numberField002>
<!--Optional:-->
<sch:numberField003>?</sch:numberField003>
<!--Optional:-->
<sch:numberField004>?</sch:numberField004>
<!--Optional:-->
<sch:numberField005>?</sch:numberField005>
<!--Optional:-->
<sch:numberField006>?</sch:numberField006>
<!--Optional:-->
<sch:numberField007>?</sch:numberField007>
<!--Optional:-->
<sch:numberField008>?</sch:numberField008>
<!--Optional:-->
<sch:numberField009>?</sch:numberField009>
<!--Optional:-->
<sch:numberField010>?</sch:numberField010>
<!--Optional:-->
<sch:numberField011>?</sch:numberField011>
<!--Optional:-->
<sch:numberField012>?</sch:numberField012>
<!--Optional:-->
<sch:numberField013>?</sch:numberField013>
<!--Optional:-->
<sch:numberField014>?</sch:numberField014>
<!--Optional:-->
<sch:numberField015>?</sch:numberField015>
<!--Optional:-->
<sch:numberField016>?</sch:numberField016>
<!--Optional:-->
<sch:numberField017>?</sch:numberField017>
<!--Optional:-->
<sch:numberField018>?</sch:numberField018>
<!--Optional:-->
<sch:numberField019>?</sch:numberField019>
<!--Optional:-->
<sch:numberField020>?</sch:numberField020>
<!--Optional:-->
<sch:rtfField001>?</sch:rtfField001>
<!--Optional:-->
<sch:rtfField002>?</sch:rtfField002>
<!--Optional:-->
```

```
                <sch:rtfField003>?</sch:rtfField003>
                <!--Optional:-->
                <sch:shortField001>?</sch:shortField001>
                <!--Optional:-->
                <sch:shortField002>?</sch:shortField002>
                <!--Optional:-->
                <sch:shortField003>?</sch:shortField003>
                <!--Optional:-->
                <sch:shortField004>?</sch:shortField004>
                <!--Optional:-->
                <sch:shortField005>?</sch:shortField005>
                <!--Optional:-->
                <sch:shortField006>?</sch:shortField006>
                <!--Optional:-->
                <sch:shortField007>?</sch:shortField007>
                <!--Optional:-->
                <sch:shortField008>?</sch:shortField008>
                <!--Optional:-->
                <sch:shortField009>?</sch:shortField009>
                <!--Optional:-->
                <sch:shortField010>?</sch:shortField010>
                <!--Optional:-->
                <sch:shortField011>?</sch:shortField011>
                <!--Optional:-->
                <sch:shortField012>?</sch:shortField012>
                <!--Optional:-->
                <sch:shortField013>?</sch:shortField013>
                <!--Optional:-->
                <sch:shortField014>?</sch:shortField014>
                <!--Optional:-->
                <sch:shortField015>?</sch:shortField015>
                <!--Optional:-->
                <sch:shortField016>?</sch:shortField016>
                <!--Optional:-->
                <sch:shortField017>?</sch:shortField017>
                <!--Optional:-->
                <sch:shortField018>?</sch:shortField018>
                <!--Optional:-->
                <sch:shortField019>?</sch:shortField019>
                <!--Optional:-->
                <sch:shortField020>?</sch:shortField020>
                <!--You may enter ANY elements at this point-->
            </sch:UserDefinedFieldsData>
         </product>
      </ext:createProductRequest>
   </soapenv:Body>
</soapenv:Envelope>
```

While most of these are self-explanatory, the following the elements need special mention:

- Envelope/Body/createProductRequest/product/ProductBarcode - These are the EAN/Barcode values to apply to the Product Record.
- Envelope/Body/createProductRequest/product/OuterBarcode - These are the Shipping Case Codes to apply to the Product Record.
- Envelope/Body/createProductRequest/product/OtherContacts - Depending on the customer setup of Brand Compliance, there can be zero to many additional roles for which users must be nominated on a Product Record. For each Role of users, a Contact sub-element is added to OtherContacts which maps to the

application's Product Record based on the Role element's value matching the Code of the user role in the application. The users are then loaded using the name supplied. For example, if the system has been set up to include the Buyer role in the Product Record's list of roles, you might see:

```
<sch:OtherContacts>
   <sch:Contact>
      <sch:Name>Joe Bloggs</sch:Name>
      <sch:Role>BUYER</sch:Role>
   </sch:Contact>
</sch:OtherContacts>
```

If a Role is specified by the system to be included on the Product Record as a Mandatory value and a value is not specified by the web service request, the user with name TBC is applied for that Role. If that user does not exist, web service calls fail with an invalid user message. If that user does exist but does not have the role required, the web service calls fail with a user does not have role error.

Calls to this service which specify a Product that already exists fail.

Response Details

If an error occurs, you see a response similar to the following:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns0:createProductResponse xmlns:ns1="http://www.micros.com/schema"
xmlns:ns0="external.service.creations.micros.com">
         <createProductResponse>
            <ns1:Status>FAILED</ns1:Status>
            <ns1:Messages>
               <ns1:Message>
                  <ns1:Severity>ERROR</ns1:Severity>
                  <ns1:Description>Specification Type FOOD does not map to a
single Specification Type Format in myCreations, product not
added.</ns1:Description>
               </ns1:Message>
            </ns1:Messages>
         </createProductResponse>
      </ns0:createProductResponse>
   </soap:Body>
</soap:Envelope>
```

If the request succeeds, you see a response like the following:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns0:createProductResponse xmlns:ns1="http://www.micros.com/schema"
xmlns:ns0="external.service.creations.micros.com">
         <createProductResponse>
            <ns1:Status>SUCCEEDED</ns1:Status>
            <ns1:Messages/>
         </createProductResponse>
      </ns0:createProductResponse>
   </soap:Body>
</soap:Envelope>
```

Even if the request succeeds, additional warning messages may be included. You should check Envelop/Body/createProductResponse/createProductResponse/Status for SUCCEEDED or FAILED to determine the success or failure of the call.

updateProduct

Description

A service to allow the update of Product Records.

Available from Version

1.8

Endpoint Address

/services/productService

Operation

updateProduct

Request Details

The request is an XML document. The following example shows the values that may be supplied:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ext="external.service.creations.micros.com"
xmlns:sch="http://www.micros.com/schema">
   <soapenv:Header/>
   <soapenv:Body>
      <ext:updateProductRequest>
         <product>
            <sch:ProductNumber>?</sch:ProductNumber>
            <sch:ProductName>?</sch:ProductName>
            <sch:SupplierCode>?</sch:SupplierCode>
            <!--Optional:-->
            <sch:SupplierName>?</sch:SupplierName>
            <!--Optional:-->
            <sch:Status>?</sch:Status>
            <!--Zero or more repetitions:-->
            <sch:ProductBarcode>?</sch:ProductBarcode>
            <!--Zero or more repetitions:-->
            <sch:OuterBarcode>?</sch:OuterBarcode>
            <!--Optional:-->
            <sch:SpecificationType>?</sch:SpecificationType>
            <!--Optional:-->
            <sch:SpecificationTypeFormat>?</sch:SpecificationTypeFormat>
            <sch:ProductTechnologist>?</sch:ProductTechnologist>
            <!--Optional:-->
            <sch:OtherContacts>
               <!--Zero or more repetitions:-->
               <sch:Contact>
                  <!--1 or more repetitions:-->
                  <sch:Name>?</sch:Name>
                  <sch:Role>?</sch:Role>
               </sch:Contact>
            </sch:OtherContacts>
            <!--Optional:-->
            <sch:ProjectName>?</sch:ProjectName>
            <sch:Quantity>?</sch:Quantity>
            <!--Optional:-->
            <sch:Sites>
               <!--Zero or more repetitions:-->
               <sch:SiteCode>?</sch:SiteCode>
```

```
</sch:Sites>
<!--Optional:-->
<sch:LeadBusinessCategory>?</sch:LeadBusinessCategory>
<!--Optional:-->
<sch:BusinessCategories>
   <!--Zero or more repetitions:-->
   <sch:Category>?</sch:Category>
</sch:BusinessCategories>
<!--Zero or more repetitions:-->
<sch:CountryWhereSold>?</sch:CountryWhereSold>
<!--Optional:-->
<sch:UserDefinedFieldsData>
   <!--Optional:-->
   <sch:dateField001>?</sch:dateField001>
   <!--Optional:-->
   <sch:dateField002>?</sch:dateField002>
   <!--Optional:-->
   <sch:dateField003>?</sch:dateField003>
   <!--Optional:-->
   <sch:dateField004>?</sch:dateField004>
   <!--Optional:-->
   <sch:dateField005>?</sch:dateField005>
   <!--Optional:-->
   <sch:dateField006>?</sch:dateField006>
   <!--Optional:-->
   <sch:dateField007>?</sch:dateField007>
   <!--Optional:-->
   <sch:dateField008>?</sch:dateField008>
   <!--Optional:-->
   <sch:dateField009>?</sch:dateField009>
   <!--Optional:-->
   <sch:dateField010>?</sch:dateField010>
   <!--Optional:-->
   <sch:dateField011>?</sch:dateField011>
   <!--Optional:-->
   <sch:dateField012>?</sch:dateField012>
   <!--Optional:-->
   <sch:dateField013>?</sch:dateField013>
   <!--Optional:-->
   <sch:dateField014>?</sch:dateField014>
   <!--Optional:-->
   <sch:dateField015>?</sch:dateField015>
   <!--Optional:-->
   <sch:dateField016>?</sch:dateField016>
   <!--Optional:-->
   <sch:dateField017>?</sch:dateField017>
   <!--Optional:-->
   <sch:dateField018>?</sch:dateField018>
   <!--Optional:-->
   <sch:dateField019>?</sch:dateField019>
   <!--Optional:-->
   <sch:dateField020>?</sch:dateField020>
   <!--Optional:-->
   <sch:longField001>?</sch:longField001>
   <!--Optional:-->
   <sch:longField002>?</sch:longField002>
   <!--Optional:-->
   <sch:longField003>?</sch:longField003>
   <!--Optional:-->
   <sch:longField004>?</sch:longField004>
```

```
<!--Optional:-->
<sch:longField005>?</sch:longField005>
<!--Optional:-->
<sch:longField006>?</sch:longField006>
<!--Optional:-->
<sch:longField007>?</sch:longField007>
<!--Optional:-->
<sch:longField008>?</sch:longField008>
<!--Optional:-->
<sch:longField009>?</sch:longField009>
<!--Optional:-->
<sch:longField010>?</sch:longField010>
<!--Optional:-->
<sch:longField011>?</sch:longField011>
<!--Optional:-->
<sch:longField012>?</sch:longField012>
<!--Optional:-->
<sch:longField013>?</sch:longField013>
<!--Optional:-->
<sch:longField014>?</sch:longField014>
<!--Optional:-->
<sch:longField015>?</sch:longField015>
<!--Optional:-->
<sch:longField016>?</sch:longField016>
<!--Optional:-->
<sch:longField017>?</sch:longField017>
<!--Optional:-->
<sch:longField018>?</sch:longField018>
<!--Optional:-->
<sch:longField019>?</sch:longField019>
<!--Optional:-->
<sch:longField020>?</sch:longField020>
<!--Optional:-->
<sch:numberField001>?</sch:numberField001>
<!--Optional:-->
<sch:numberField002>?</sch:numberField002>
<!--Optional:-->
<sch:numberField003>?</sch:numberField003>
<!--Optional:-->
<sch:numberField004>?</sch:numberField004>
<!--Optional:-->
<sch:numberField005>?</sch:numberField005>
<!--Optional:-->
<sch:numberField006>?</sch:numberField006>
<!--Optional:-->
<sch:numberField007>?</sch:numberField007>
<!--Optional:-->
<sch:numberField008>?</sch:numberField008>
<!--Optional:-->
<sch:numberField009>?</sch:numberField009>
<!--Optional:-->
<sch:numberField010>?</sch:numberField010>
<!--Optional:-->
<sch:numberField011>?</sch:numberField011>
<!--Optional:-->
<sch:numberField012>?</sch:numberField012>
<!--Optional:-->
<sch:numberField013>?</sch:numberField013>
<!--Optional:-->
<sch:numberField014>?</sch:numberField014>
```

```
      <!--Optional:-->
      <sch:numberField015>?</sch:numberField015>
      <!--Optional:-->
      <sch:numberField016>?</sch:numberField016>
      <!--Optional:-->
      <sch:numberField017>?</sch:numberField017>
      <!--Optional:-->
      <sch:numberField018>?</sch:numberField018>
      <!--Optional:-->
      <sch:numberField019>?</sch:numberField019>
      <!--Optional:-->
      <sch:numberField020>?</sch:numberField020>
      <!--Optional:-->
      <sch:rtfField001>?</sch:rtfField001>
      <!--Optional:-->
      <sch:rtfField002>?</sch:rtfField002>
      <!--Optional:-->
      <sch:rtfField003>?</sch:rtfField003>
      <!--Optional:-->
      <sch:shortField001>?</sch:shortField001>
      <!--Optional:-->
      <sch:shortField002>?</sch:shortField002>
      <!--Optional:-->
      <sch:shortField003>?</sch:shortField003>
      <!--Optional:-->
      <sch:shortField004>?</sch:shortField004>
      <!--Optional:-->
      <sch:shortField005>?</sch:shortField005>
      <!--Optional:-->
      <sch:shortField006>?</sch:shortField006>
      <!--Optional:-->
      <sch:shortField007>?</sch:shortField007>
      <!--Optional:-->
      <sch:shortField008>?</sch:shortField008>
      <!--Optional:-->
      <sch:shortField009>?</sch:shortField009>
      <!--Optional:-->
      <sch:shortField010>?</sch:shortField010>
      <!--Optional:-->
      <sch:shortField011>?</sch:shortField011>
      <!--Optional:-->
      <sch:shortField012>?</sch:shortField012>
      <!--Optional:-->
      <sch:shortField013>?</sch:shortField013>
      <!--Optional:-->
      <sch:shortField014>?</sch:shortField014>
      <!--Optional:-->
      <sch:shortField015>?</sch:shortField015>
      <!--Optional:-->
      <sch:shortField016>?</sch:shortField016>
      <!--Optional:-->
      <sch:shortField017>?</sch:shortField017>
      <!--Optional:-->
      <sch:shortField018>?</sch:shortField018>
      <!--Optional:-->
      <sch:shortField019>?</sch:shortField019>
      <!--Optional:-->
      <sch:shortField020>?</sch:shortField020>
      <!--You may enter ANY elements at this point-->
   </sch:UserDefinedFieldsData>
```

```
         </product>
      </ext:updateProductRequest>
   </soapenv:Body>
</soapenv:Envelope>
```

This is basically the same as for the createProduct operation. Refer to the createProduct documentation for more details.

In order to update a product, it must already exist. Calls to this service for products that do not exist fail.

Response Details

If an error occurs, you see a response similar to the following:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns0:updateProductResponse xmlns:ns1="http://www.micros.com/schema"
xmlns:ns0="external.service.creations.micros.com">
         <updateProductResponse>
            <ns1:Status>FAILED</ns1:Status>
            <ns1:Messages>
               <ns1:Message>
                  <ns1:Severity>ERROR</ns1:Severity>
                  <ns1:Description>Specification Type FOOD does not map to a
single Specification Type Format in myCreations, product not
added.</ns1:Description>
               </ns1:Message>
            </ns1:Messages>
         </updateProductResponse>
      </ns0:updateProductResponse>
   </soap:Body>
</soap:Envelope>
```

If the request succeeds, you see a response like the following:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns0:updateProductResponse xmlns:ns1="http://www.micros.com/schema"
xmlns:ns0="external.service.creations.micros.com">
         <updateProductResponse>
            <ns1:Status>SUCCEEDED</ns1:Status>
            <ns1:Messages/>
         </updateProductResponse>
      </ns0:updateProductResponse>
   </soap:Body>
</soap:Envelope>
```

Even if the request succeeds, additional warning messages may be included. You should check
Envelop/Body/updateProductResponse/updateProductResponse/Status for SUCCEEDED or FAILED to determine the success or failure of the call.

submitProduct

Description

A service to allow the update and creation of Product Records.

Available from Version

1.8

Endpoint Address

/services/productService

Operation

submitProduct

Request Details

The request is an XML document. The following example shows the values that may be supplied:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ext="external.service.creations.micros.com"
xmlns:sch="http://www.micros.com/schema">
   <soapenv:Header/>
   <soapenv:Body>
      <ext:submitProductRequest>
         <product>
            <sch:ProductNumber>?</sch:ProductNumber>
            <sch:ProductName>?</sch:ProductName>
            <sch:SupplierCode>?</sch:SupplierCode>
            <!--Optional:-->
            <sch:SupplierName>?</sch:SupplierName>
            <!--Optional:-->
            <sch:Status>?</sch:Status>
            <!--Zero or more repetitions:-->
            <sch:ProductBarcode>?</sch:ProductBarcode>
            <!--Zero or more repetitions:-->
            <sch:OuterBarcode>?</sch:OuterBarcode>
            <!--Optional:-->
            <sch:SpecificationType>?</sch:SpecificationType>
            <!--Optional:-->
            <sch:SpecificationTypeFormat>?</sch:SpecificationTypeFormat>
            <sch:ProductTechnologist>?</sch:ProductTechnologist>
            <!--Optional:-->
            <sch:OtherContacts>
               <!--Zero or more repetitions:-->
               <sch:Contact>
                  <!--1 or more repetitions:-->
                  <sch:Name>?</sch:Name>
                  <sch:Role>?</sch:Role>
               </sch:Contact>
            </sch:OtherContacts>
            <!--Optional:-->
            <sch:ProjectName>?</sch:ProjectName>
            <sch:Quantity>?</sch:Quantity>
            <!--Optional:-->
            <sch:Sites>
               <!--Zero or more repetitions:-->
               <sch:SiteCode>?</sch:SiteCode>
            </sch:Sites>
            <!--Optional:-->
            <sch:LeadBusinessCategory>?</sch:LeadBusinessCategory>
            <!--Optional:-->
            <sch:BusinessCategories>
               <!--Zero or more repetitions:-->
```

```
      <sch:Category>?</sch:Category>
   </sch:BusinessCategories>
   <!--Zero or more repetitions:-->
   <sch:CountryWhereSold>?</sch:CountryWhereSold>
   <!--Optional:-->
   <sch:UserDefinedFieldsData>
      <!--Optional:-->
      <sch:dateField001>?</sch:dateField001>
      <!--Optional:-->
      <sch:dateField002>?</sch:dateField002>
      <!--Optional:-->
      <sch:dateField003>?</sch:dateField003>
      <!--Optional:-->
      <sch:dateField004>?</sch:dateField004>
      <!--Optional:-->
      <sch:dateField005>?</sch:dateField005>
      <!--Optional:-->
      <sch:dateField006>?</sch:dateField006>
      <!--Optional:-->
      <sch:dateField007>?</sch:dateField007>
      <!--Optional:-->
      <sch:dateField008>?</sch:dateField008>
      <!--Optional:-->
      <sch:dateField009>?</sch:dateField009>
      <!--Optional:-->
      <sch:dateField010>?</sch:dateField010>
      <!--Optional:-->
      <sch:dateField011>?</sch:dateField011>
      <!--Optional:-->
      <sch:dateField012>?</sch:dateField012>
      <!--Optional:-->
      <sch:dateField013>?</sch:dateField013>
      <!--Optional:-->
      <sch:dateField014>?</sch:dateField014>
      <!--Optional:-->
      <sch:dateField015>?</sch:dateField015>
      <!--Optional:-->
      <sch:dateField016>?</sch:dateField016>
      <!--Optional:-->
      <sch:dateField017>?</sch:dateField017>
      <!--Optional:-->
      <sch:dateField018>?</sch:dateField018>
      <!--Optional:-->
      <sch:dateField019>?</sch:dateField019>
      <!--Optional:-->
      <sch:dateField020>?</sch:dateField020>
      <!--Optional:-->
      <sch:longField001>?</sch:longField001>
      <!--Optional:-->
      <sch:longField002>?</sch:longField002>
      <!--Optional:-->
      <sch:longField003>?</sch:longField003>
      <!--Optional:-->
      <sch:longField004>?</sch:longField004>
      <!--Optional:-->
      <sch:longField005>?</sch:longField005>
      <!--Optional:-->
      <sch:longField006>?</sch:longField006>
      <!--Optional:-->
      <sch:longField007>?</sch:longField007>
```

```
<!--Optional:-->
<sch:longField008>?</sch:longField008>
<!--Optional:-->
<sch:longField009>?</sch:longField009>
<!--Optional:-->
<sch:longField010>?</sch:longField010>
<!--Optional:-->
<sch:longField011>?</sch:longField011>
<!--Optional:-->
<sch:longField012>?</sch:longField012>
<!--Optional:-->
<sch:longField013>?</sch:longField013>
<!--Optional:-->
<sch:longField014>?</sch:longField014>
<!--Optional:-->
<sch:longField015>?</sch:longField015>
<!--Optional:-->
<sch:longField016>?</sch:longField016>
<!--Optional:-->
<sch:longField017>?</sch:longField017>
<!--Optional:-->
<sch:longField018>?</sch:longField018>
<!--Optional:-->
<sch:longField019>?</sch:longField019>
<!--Optional:-->
<sch:longField020>?</sch:longField020>
<!--Optional:-->
<sch:numberField001>?</sch:numberField001>
<!--Optional:-->
<sch:numberField002>?</sch:numberField002>
<!--Optional:-->
<sch:numberField003>?</sch:numberField003>
<!--Optional:-->
<sch:numberField004>?</sch:numberField004>
<!--Optional:-->
<sch:numberField005>?</sch:numberField005>
<!--Optional:-->
<sch:numberField006>?</sch:numberField006>
<!--Optional:-->
<sch:numberField007>?</sch:numberField007>
<!--Optional:-->
<sch:numberField008>?</sch:numberField008>
<!--Optional:-->
<sch:numberField009>?</sch:numberField009>
<!--Optional:-->
<sch:numberField010>?</sch:numberField010>
<!--Optional:-->
<sch:numberField011>?</sch:numberField011>
<!--Optional:-->
<sch:numberField012>?</sch:numberField012>
<!--Optional:-->
<sch:numberField013>?</sch:numberField013>
<!--Optional:-->
<sch:numberField014>?</sch:numberField014>
<!--Optional:-->
<sch:numberField015>?</sch:numberField015>
<!--Optional:-->
<sch:numberField016>?</sch:numberField016>
<!--Optional:-->
<sch:numberField017>?</sch:numberField017>
```

```
                    <!--Optional:-->
                    <sch:numberField018>?</sch:numberField018>
                    <!--Optional:-->
                    <sch:numberField019>?</sch:numberField019>
                    <!--Optional:-->
                    <sch:numberField020>?</sch:numberField020>
                    <!--Optional:-->
                    <sch:rtfField001>?</sch:rtfField001>
                    <!--Optional:-->
                    <sch:rtfField002>?</sch:rtfField002>
                    <!--Optional:-->
                    <sch:rtfField003>?</sch:rtfField003>
                    <!--Optional:-->
                    <sch:shortField001>?</sch:shortField001>
                    <!--Optional:-->
                    <sch:shortField002>?</sch:shortField002>
                    <!--Optional:-->
                    <sch:shortField003>?</sch:shortField003>
                    <!--Optional:-->
                    <sch:shortField004>?</sch:shortField004>
                    <!--Optional:-->
                    <sch:shortField005>?</sch:shortField005>
                    <!--Optional:-->
                    <sch:shortField006>?</sch:shortField006>
                    <!--Optional:-->
                    <sch:shortField007>?</sch:shortField007>
                    <!--Optional:-->
                    <sch:shortField008>?</sch:shortField008>
                    <!--Optional:-->
                    <sch:shortField009>?</sch:shortField009>
                    <!--Optional:-->
                    <sch:shortField010>?</sch:shortField010>
                    <!--Optional:-->
                    <sch:shortField011>?</sch:shortField011>
                    <!--Optional:-->
                    <sch:shortField012>?</sch:shortField012>
                    <!--Optional:-->
                    <sch:shortField013>?</sch:shortField013>
                    <!--Optional:-->
                    <sch:shortField014>?</sch:shortField014>
                    <!--Optional:-->
                    <sch:shortField015>?</sch:shortField015>
                    <!--Optional:-->
                    <sch:shortField016>?</sch:shortField016>
                    <!--Optional:-->
                    <sch:shortField017>?</sch:shortField017>
                    <!--Optional:-->
                    <sch:shortField018>?</sch:shortField018>
                    <!--Optional:-->
                    <sch:shortField019>?</sch:shortField019>
                    <!--Optional:-->
                    <sch:shortField020>?</sch:shortField020>
                    <!--You may enter ANY elements at this point-->
                </sch:UserDefinedFieldsData>
            </product>
        </ext:submitProductRequest>
    </soapenv:Body>
</soapenv:Envelope>
```

This is basically the same as for the createProduct and updateProduct operations. Refer to the documentation for those operations for more details.

This service determines if the specified product already exists and if so, updates it. Otherwise, it creates the product.

Response Details

If an error occurs, you see a response similar to the following:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns0:submitProductResponse xmlns:ns1="http://www.micros.com/schema"
xmlns:ns0="external.service.creations.micros.com">
         <submitProductResponse>
            <ns1:Status>FAILED</ns1:Status>
            <ns1:Messages>
               <ns1:Message>
                  <ns1:Severity>ERROR</ns1:Severity>
                  <ns1:Description>Specification Type FOOD does not map to a
single Specification Type Format in myCreations, product not
added.</ns1:Description>
               </ns1:Message>
            </ns1:Messages>
         </submitProductResponse>
      </ns0:submitProductResponse>
   </soap:Body>
</soap:Envelope>
```

If the request succeeds, you see a response similar to the following:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns0:submitProductResponse xmlns:ns1="http://www.micros.com/schema"
xmlns:ns0="external.service.creations.micros.com">
         <submitProductResponse>
            <ns1:Status>SUCCEEDED</ns1:Status>
            <ns1:Messages/>
         </submitProductResponse>
      </ns0:submitProductResponse>
   </soap:Body>
</soap:Envelope>
```

Even if the request succeeds, additional warning messages may be included. You should check
Envelop/Body/submitProductResponse/submitProductResponse/Status for SUCCEEDED or FAILED to determine the success or failure of the call.

myArtwork Activities

This section describes the API for project activities.

Activity Management

Description

A service to allow updates to Activities from an external service. Used in myArtwork integration.

Available from Version
1.8

Endpoint Address
/services/activityManagement

Operation
updateActivityStatus

Request Details
The request is an XML document. The following is an example:

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ext="external.service.creations.micros.com">
   <soapenv:Header/>
   <soapenv:Body>
      <ext:updateActivityStatus>
         <!--Optional:-->
         <activityUpdateRequest>
            <!--Optional:-->
            <activityName>?</activityName>
            <activityRecordId>?</activityRecordId>
            <!--Optional:-->
            <projectId>?</projectId>
            <projectRecordId>?</projectRecordId>
            <subStatusCode>?</subStatusCode>
         </activityUpdateRequest>
      </ext:updateActivityStatus>
   </soapenv:Body>
</soapenv:Envelope>
```

The activityRecordId refers to the internal ID of the activity, productRecordId refers to the internal ID of the project that the activity exists on, and subStatusCode is the CODE of the Activity Sub-Status listed, in the Activity's list of allowable sub-statuses, to which the activity's sub status is changed.

Response Details
If an error occurs, you see a response similar to the following:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
      <ns0:updateActivityStatusResponse
xmlns:ns0="external.service.creations.micros.com">
         <return>SUCCESS</return>
      </ns0:updateActivityStatusResponse>
   </soap:Body>
</soap:Envelope>
```

In the event of values not being provided, the return element contains one of the following:

- FAILED - No activityRecordId provided
- FAILED - No projectRecordId provided
- FAILED - No subStatusCode provided

If there is an internal failure in creating the background job to perform the update, the return element contains the following:

FAILED

4

RESTful APIs

The following RESTful APIs are available:

- UserRestService - Users
- SupplierRestService - Suppliers
- SiteRestService - Sites
- ContactRestService - Supplier/Site Contacts
- ProductRecordRestService - Product Records
- ProductSpecificationRestService - Product Specifications
- TaskRestService - Users' Tasks
- UrgentItemsRestService - Users' Urgent Items
- DataPrivacyService: Data Privacy Requests

UserRestService

This section describes the API for managing users.

List of Values

Description

Retrieves a list of users in a paged list.

Available from Version

1.9

Endpoint Address

/services/rest/user

HTTP Method

GET

Request Parameters

Parameters are passed as URI parameters.

URI Parameters

Parameter Name	Mandatory/Optional	Value Type	Example
offset	Optional	Numeric	60
pageSize	Optional	Numeric	30

The offset and pageSize parameters control the paging of the list. The offset defines the first record to retrieve (0-based) and the pageSize defines how many records to include in the list.

Response

For a successful response, XML is returned with a UserLinkList root element containing an entries element for each matched user with the recordId, loginId, name, and a recordLink URI to the UserRestService Retrieve record service for this user. For example:

```
<UserLinkList>
    <entries xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="userLink">
        <recordId>1</recordId>
        <recordLink>
            http://localhost:9090/creations-salesdemo/services/rest/user/1
        </recordLink>
        <loginId>root</loginId>
        <name>Super User</name>
    </entries>
    <previousPage>
http://localhost:9090/creations-salesdemo/services/rest/user/?offset=0&amp;pageSiz
e=30
    </previousPage>
    <nextPage>
http://localhost:9090/creations-salesdemo/services/rest/user/?offset=60&amp;pageSi
ze=30
    </nextPage>
    <totalRecords>154</totalRecords>
    <entryArray>
        <recordId>1</recordId>
        <recordLink>
            http://localhost:9090/creations-salesdemo/services/rest/user/1
        </recordLink>
        <loginId>root</loginId>
        <name>Super User</name>
    </entryArray>
</UserLinkList>
```

Error Responses

In the event that an error occurs, an HTTP 500 response is sent.

Retrieve Record

Description

Retrieves a single user's details.

Available from Version

1.9

Endpoint Address

/services/rest/user/{id}

HTTP Method

GET

Request Parameters

There are no request parameters, but the URL contains the {id} parameter that determines the record to load.

Response

For a successful response, XML is returned with a userFullDTO root element containing the details of the user requested. For example:

```
<ns0:userFullDTO
    xmlns:ns1="http://www.micros.com/creations/core/domain/dto/v1p0/simple"
    xmlns:ns0="http://www.micros.com/creations/core/domain/dto/v1p0/full">
    <ns0:areas>
        <ns1:code>AREA_D</ns1:code>
        <ns1:localeData>
            <ns0:description><![CDATA[Area 51]]></ns0:description>
            <ns0:id>2</ns0:id>
        </ns1:localeData>
        <ns1:id>2</ns1:id>
    </ns0:areas>
    <ns0:areas>
        <ns1:code>AREA_B</ns1:code>
        <ns1:localeData>
            <ns0:description><![CDATA[Preview]]></ns0:description>
            <ns0:id>4</ns0:id>
        </ns1:localeData>
        <ns1:id>4</ns1:id>
    </ns0:areas>
    <ns0:areas>
        <ns1:code>AREA_C</ns1:code>
        <ns1:localeData>
            <ns0:description><![CDATA[Test]]></ns0:description>
            <ns0:id>3</ns0:id>
        </ns1:localeData>
        <ns1:id>3</ns1:id>
    </ns0:areas>
    <ns0:code>techadmin</ns0:code>
    <ns0:disabled>false</ns0:disabled>
    <ns0:person>
        <ns0:address>
            <ns0:id>55</ns0:id>
        </ns0:address>
        <ns0:contactDetails>
            <ns0:id>54</ns0:id>
        </ns0:contactDetails>
        <ns0:email>techadmin@example.com</ns0:email>
        <ns0:jobTitle>Administrator</ns0:jobTitle>
        <ns0:language>
            <ns1:code>en_GB</ns1:code>
            <ns1:isActive>true</ns1:isActive>
```

```
            <ns1:localeData>
                <ns0:description><![CDATA[English (British)]]></ns0:description>
                <ns0:id>10</ns0:id>
            </ns1:localeData>
            <ns1:id>2</ns1:id>
        </ns0:language>
        <ns0:name>Technical Admin</ns0:name>
        <ns0:supplier>
            <ns1:code>RETAILER</ns1:code>
            <ns1:companyNumber>12345</ns1:companyNumber>
            <ns1:email>blackhole@example.com</ns1:email>
            <ns1:isActive>true</ns1:isActive>
            <ns1:name>Oracle Retail Brand Compliance Management</ns1:name>
            <ns1:supplierCodeConfirmed>false</ns1:supplierCodeConfirmed>
            <ns1:supplierContactName>Joe Bloggs</ns1:supplierContactName>
            <ns1:id>1</ns1:id>
        </ns0:supplier>
        <ns0:id>2</ns0:id>
    </ns0:person>
    <ns0:role>
        <ns1:code>PRODUCT TECHNOLOGIST</ns1:code>
        <ns1:localeData>
            <ns0:description><![CDATA[Product Technologist]]></ns0:description>
            <ns0:id>1</ns0:id>
        </ns1:localeData>
        <ns1:id>1</ns1:id>
    </ns0:role>
    <ns0:supplier>
        <ns1:code>RETAILER</ns1:code>
        <ns1:companyNumber>12345</ns1:companyNumber>
        <ns1:email>blackhole@example.com</ns1:email>
        <ns1:isActive>true</ns1:isActive>
        <ns1:name>Oracle Retail Brand Compliance Management</ns1:name>
        <ns1:supplierCodeConfirmed>false</ns1:supplierCodeConfirmed>
        <ns1:supplierContactName>Joe Bloggs</ns1:supplierContactName>
        <ns1:id>1</ns1:id>
    </ns0:supplier>
    <ns0:timeZone>Europe/London</ns0:timeZone>
    <ns0:udfData>
        <ns0:id>3</ns0:id>
    </ns0:udfData>
    <ns0:userType>RETAILER</ns0:userType>
    <ns0:id>2</ns0:id>
</ns0:userFullDTO>
```

Error Responses

In the event that an error occurs, an HTTP 500 response is sent.

Check Record Modification Timestamp

Description

Retrieves the last modification time for a user.

Available from Version

1.9

Endpoint Address
/services/rest/user/{id}

HTTP Method
HEAD

Request Parameters
There are no request parameters, but the URL contains the {id} parameter that determines the record to load.

Response
If successful, an HTTP 200 response is sent containing the Last-Modified header to show the last modification data for that user. For example:

```
HTTP/1.1 200 OK
Content-Length: 0
Expires: Thu, 01-Jan-1970 00:00:00 GMT
Set-Cookie: JSESSIONID=154osm4djwouv;Path=/creations-salesdemo
Date: Thu, 30 Apr 2015 10:33:05 GMT
Last-Modified: Fri, 28 Feb 2014 16:27:06 GMT
Server: Jetty(7.0.2.v20100331)
```

Error Responses
In the event that an error occurs, an HTTP 500 response is sent.

Create Record

Description
Creates a new user.

Available from Version
1.9

Endpoint Address
/services/rest/user

HTTP Method
POST

Request Body
The body of the request contains a UserFullDTO to specify the user to create. For example:

```
<ns0:userFullDTO
xmlns:ns1="http://www.micros.com/creations/core/domain/dto/v1p0/simple"
xmlns:ns0="http://www.micros.com/creations/core/domain/dto/v1p0/full">
   <ns0:code>testuser</ns0:code>
   <ns0:disabled>false</ns0:disabled>
   <ns0:person>
      <ns0:email>testuser@example.com</ns0:email>
      <ns0:jobTitle>Administrator</ns0:jobTitle>
      <ns0:language>
         <ns1:code>en_GB</ns1:code>
      </ns0:language>
```

```
      <ns0:name>testuser</ns0:name>
      <ns0:supplier>
         <ns1:code>RETAILER</ns1:code>
      </ns0:supplier>
   </ns0:person>
   <ns0:role>
      <ns1:code>PRODUCT TECHNOLOGIST</ns1:code>
   </ns0:role>
   <ns0:supplier>
      <ns1:code>RETAILER</ns1:code>
   </ns0:supplier>
   <ns0:timeZone>Europe/London</ns0:timeZone>
   <ns0:userType>RETAILER</ns0:userType>
</ns0:userFullDTO>
```

Compared to the response for retrieving a user (which uses the same UserFullDTO type), this request is much shorter. This is achieved by only including the information relevant to creating the user. Where the record is linked to another record, such as the role, only the business keys (the code in this case) that allow that linked record to be identified are supplied. In general, this is the code attribute of the linked record, but in some cases, it is the status or a combination of values.

Response

If successful, an HTTP 200 response is sent with a body containing a UserLink:

```
<UserLink>
   <recordId>158</recordId>
   <recordLink>
      http://localhost:9090/creations-salesdemo/services/rest/user/158
   </recordLink>
   <loginId>testuser</loginId>
   <name>testuser</name>
</UserLink>
```

The code of the request is mapped to the loginId of the response and the person/name is mapped to the name. The recordId is the ID of the newly created user and the recordLink is a URI to the user record that was created for use in a GET request.

Error Responses

If the supplied data does not result in a valid User (such as the login ID missing, no roles or authority profiles assigned), an HTTP 417 response is sent with a body message stating the validation errors. The request should not be reattempted with the same content.

If an internal error occurs, an HTTP 500 response is sent.

Update Record

Description

Updates an existing user.

Available from Version

1.9

Endpoint Address

/services/rest/user/{id}

HTTP Method

PUT

Request Body

The body of the request contains a UserFullDTO to specify how the user should appear after the update:

```
<ns0:userFullDTO
xmlns:ns1="http://www.micros.com/creations/core/domain/dto/v1p0/simple"
xmlns:ns0="http://www.micros.com/creations/core/domain/dto/v1p0/full">
   <ns0:code>testuser</ns0:code>
   <ns0:disabled>false</ns0:disabled>
   <ns0:person>
      <ns0:email>testuser@example.com</ns0:email>
      <ns0:jobTitle>Administrator</ns0:jobTitle>
      <ns0:language>
         <ns1:code>en_GB</ns1:code>
      </ns0:language>
      <ns0:name>testuser</ns0:name>
      <ns0:supplier>
         <ns1:code>RETAILER</ns1:code>
      </ns0:supplier>
   </ns0:person>
   <ns0:role>
      <ns1:code>PRODUCT TECHNOLOGIST</ns1:code>
   </ns0:role>
   <ns0:supplier>
      <ns1:code>RETAILER</ns1:code>
   </ns0:supplier>
   <ns0:timeZone>Europe/London</ns0:timeZone>
   <ns0:userType>RETAILER</ns0:userType>
</ns0:userFullDTO>
```

Compared to the response for retrieving a user (which uses the same UserFullDTO type), this request is much shorter. This is achieved by only including the information relevant to creating the user. Where the record is linked to another record, such as the role, only the business keys (the code in this case) that allow that linked record to be identified are supplied. In general, this is the code attribute of the linked record, but in some cases, it is the status or a combination of values.

Response

If successful, an HTTP 200 response is sent with a body containing a UserLink:

```
<UserLink>
   <recordId>158</recordId>
   <recordLink>
      http://localhost:9090/creations-salesdemo/services/rest/user/158
   </recordLink>
   <loginId>testuser</loginId>
   <name>testuser</name>
</UserLink>
```

The code of the request is mapped to the loginId of the response and the person/name is mapped to the name. The recordId is the ID of the newly created user and the recordLink is a URI to the user record that was created for use in a GET request.

Error Responses

If the supplied data does not result in a valid user (such as the login ID missing or no roles or authority profiles assigned), an HTTP 417 response is sent with a body message stating the validation errors. The request should not be reattempted with the same content.

If an internal error occurs, an HTTP 500 response is sent.

SupplierRestService

This section describes the API for managing suppliers.

List of Values

Description

Retrieves a list of suppliers in a paged list.

Available from Version

1.9

Endpoint Address

/services/rest/supplier

HTTP Method

GET

Request Parameters

Parameters are passed as URI parameters.

URI Parameters

Parameter Name	Mandatory/Optional	Value Type	Example
offset	Optional	Numeric	60
pageSize	Optional	Numeric	30

The offset and pageSize parameters control the paging of the list. The offset defines the first record to retrieve (0-based) and the pageSize defines how many records to include in the list.

Response

For a successful response, XML is returned with a SupplierLinkList root element containing an entries element for each matched supplier with the recordId, code, name, and a recordLink URI to the SupplierRestService Retrieve record service for this supplier. For example:

```
<SupplierLinkList>
    <entries xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="userLink">
        <recordId>2</recordId>
        <recordLink>
            http://localhost:9090/creations-salesdemo/services/rest/supplier/2
        </recordLink>
```

```
        <code>A0001</code>
        <name>Example Supplier</name>
    </entries>
    <previousPage>
http://localhost:9090/creations-salesdemo/services/rest/supplier/?offset=0&amp;pag
eSize=30
    </previousPage>
    <nextPage>
http://localhost:9090/creations-salesdemo/services/rest/supplier/?offset=60&amp;pa
geSize=30
    </nextPage>
    <totalRecords>154</totalRecords>
    <entryArray>
        <recordId>2</recordId>
        <recordLink>
            http://localhost:9090/creations-salesdemo/services/rest/supplier/2
        </recordLink>
        <code>A0001</code>
        <name>Example Supplier</name>
    </entryArray>
</SupplierLinkList>
```

Error Responses

In the event that an error occurs, an HTTP 500 response is sent.

Retrieve Record

Description

Retrieves a single supplier's details.

Available from Version

1.9

Endpoint Address

/services/rest/supplier/{id}

HTTP Method

GET

Request Parameters

There are no request parameters, but the URL contains the {id} parameter that determines the record to load.

Response

For a successful response, XML is returned with a supplierFullDTO root element containing the details of the supplier requested. For example:

```
<ns0:supplierFullDTO
    xmlns:ns1="http://www.micros.com/creations/core/domain/dto/v1p0/simple"
    xmlns:ns0="http://www.micros.com/creations/core/domain/dto/v1p0/full">
    <ns0:billingAddress>
        <ns0:country>
            <ns1:code>IE</ns1:code>
            <ns1:countryType>COUNTRY</ns1:countryType>
            <ns1:localeData>
```

```
                <ns0:description><![CDATA[Ireland]]></ns0:description>
                <ns0:id>3375</ns0:id>
            </ns1:localeData>
            <ns1:id>228</ns1:id>
        </ns0:country>
        <ns0:line1>road</ns0:line1>
        <ns0:line2>Town</ns0:line2>
        <ns0:line4>Area</ns0:line4>
        <ns0:postalCode>Postal Code</ns0:postalCode>
        <ns0:id>74</ns0:id>
    </ns0:billingAddress>
    <ns0:billingAddressTo>Amber Example</ns0:billingAddressTo>
    <ns0:billingCode>
        <ns1:chargePerSite>250</ns1:chargePerSite>
        <ns1:code>MEDIUM</ns1:code>
        <ns1:fixedPrice>1500</ns1:fixedPrice>
        <ns1:localeData>
            <ns0:description><![CDATA[medium]]></ns0:description>
            <ns0:id>2</ns0:id>
        </ns1:localeData>
        <ns1:id>2</ns1:id>
    </ns0:billingCode>
    <ns0:billingContact>Amber Example</ns0:billingContact>
    <ns0:billingEmail>aexample@example.com</ns0:billingEmail>
    <ns0:billingFax>0123456798</ns0:billingFax>
    <ns0:billingPhone>012364597</ns0:billingPhone>
    <ns0:businessUnit>
        <ns1:code>UK</ns1:code>
        <ns1:localeData>
            <ns0:description><![CDATA[UK]]></ns0:description>
            <ns0:id>4</ns0:id>
        </ns1:localeData>
        <ns1:id>4</ns1:id>
    </ns0:businessUnit>
    <ns0:code>A0001</ns0:code>
    <ns0:companyNumber>123981</ns0:companyNumber>
    <ns0:contactDetails>
        <ns0:fax>012346579</ns0:fax>
        <ns0:phoneNumber>0123645987</ns0:phoneNumber>
        <ns0:id>73</ns0:id>
    </ns0:contactDetails>
    <ns0:dateLastVerified>2014-02-21</ns0:dateLastVerified>
    <ns0:email>aexample@example.com</ns0:email>
    <ns0:isActive>true</ns0:isActive>
    <ns0:mainAddress>
        <ns0:country>
            <ns1:code>GB</ns1:code>
            <ns1:countryType>COUNTRY</ns1:countryType>
            <ns1:internetDomainSuffix>co.uk</ns1:internetDomainSuffix>
            <ns1:localeData>
                <ns0:description><![CDATA[United Kingdom]]></ns0:description>
                <ns0:id>553</ns0:id>
            </ns1:localeData>
            <ns1:id>38</ns1:id>
        </ns0:country>
        <ns0:line1>House name</ns0:line1>
        <ns0:line2>Road</ns0:line2>
        <ns0:line3>City</ns0:line3>
        <ns0:line4>County</ns0:line4>
        <ns0:line4Local>Local County</ns0:line4Local>
```

```
    <ns0:postalCode>POSTALCODE</ns0:postalCode>
    <ns0:id>75</ns0:id>
</ns0:mainAddress>
<ns0:name>AAA FOOD</ns0:name>
<ns0:site>
    <ns1:code>A0001-0003</ns1:code>
    <ns1:dateLastVerified>2014-02-21</ns1:dateLastVerified>
    <ns1:name>Produce</ns1:name>
    <ns1:id>2</ns1:id>
</ns0:site>
<ns0:site>
    <ns1:code>A0001-0002</ns1:code>
    <ns1:dateLastVerified>2014-02-21</ns1:dateLastVerified>
    <ns1:name>CNF</ns1:name>
    <ns1:id>4</ns1:id>
</ns0:site>
<ns0:site>
    <ns1:code>A0001-0004</ns1:code>
    <ns1:dateLastVerified>2014-02-21</ns1:dateLastVerified>
    <ns1:name>Food</ns1:name>
    <ns1:id>3</ns1:id>
</ns0:site>
<ns0:site>
    <ns1:code>A0001-0001</ns1:code>
    <ns1:dateLastVerified>2014-02-21</ns1:dateLastVerified>
    <ns1:name>FNF</ns1:name>
    <ns1:id>1</ns1:id>
</ns0:site>
<ns0:status>REGISTERED</ns0:status>
<ns0:supplierCodeConfirmed>true</ns0:supplierCodeConfirmed>
<ns0:supplierContactName>Amber Example</ns0:supplierContactName>
<ns0:supplierContacts>
    <ns0:address>
        <ns0:country>
            <ns1:code>GB</ns1:code>
            <ns1:countryType>COUNTRY</ns1:countryType>
            <ns1:internetDomainSuffix>co.uk</ns1:internetDomainSuffix>
            <ns1:localeData>
                <ns0:description><![CDATA[United Kingdom]]></ns0:description>
                <ns0:id>553</ns0:id>
            </ns1:localeData>
            <ns1:id>38</ns1:id>
        </ns0:country>
        <ns0:line1>House name</ns0:line1>
        <ns0:line2>Road</ns0:line2>
        <ns0:line3>City</ns0:line3>
        <ns0:line4>County</ns0:line4>
        <ns0:line4Local>Local County</ns0:line4Local>
        <ns0:postalCode>POSTALCODE</ns0:postalCode>
        <ns0:id>86</ns0:id>
    </ns0:address>
    <ns0:company>
        <ns1:billingAddressTo>Amber Example</ns1:billingAddressTo>
        <ns1:billingContact>Amber Example</ns1:billingContact>
        <ns1:billingEmail>aexample@example.com</ns1:billingEmail>
        <ns1:billingFax>0123456798</ns1:billingFax>
        <ns1:billingPhone>0123654987</ns1:billingPhone>
        <ns1:code>A0001</ns1:code>
        <ns1:companyNumber>123981</ns1:companyNumber>
        <ns1:dateLastVerified>2014-02-21</ns1:dateLastVerified>
```

```
                <ns1:email>aexample@example.com</ns1:email>
                <ns1:isActive>true</ns1:isActive>
                <ns1:name>AAA FOOD</ns1:name>
                <ns1:status>REGISTERED</ns1:status>
                <ns1:supplierCodeConfirmed>true</ns1:supplierCodeConfirmed>
                <ns1:supplierContactName>Amber Example</ns1:supplierContactName>
                <ns1:vatNumber>545274</ns1:vatNumber>
                <ns1:id>2</ns1:id>
            </ns0:company>
            <ns0:contactDetails>
                <ns0:fax>0123465798</ns0:fax>
                <ns0:phoneNumber>0132654987</ns0:phoneNumber>
                <ns0:id>79</ns0:id>
            </ns0:contactDetails>
            <ns0:dtype>SupplierContact</ns0:dtype>
            <ns0:person>
                <ns1:email>aexample@example.com</ns1:email>
                <ns1:name>Amber Example</ns1:name>
                <ns1:id>21</ns1:id>
            </ns0:person>
            <ns0:site>
                <ns1:code>A0001-0002</ns1:code>
                <ns1:dateLastVerified>2014-02-21</ns1:dateLastVerified>
                <ns1:name>CNF</ns1:name>
                <ns1:id>4</ns1:id>
            </ns0:site>
            <ns0:site>
                <ns1:code>A0001-0001</ns1:code>
                <ns1:dateLastVerified>2014-02-21</ns1:dateLastVerified>
                <ns1:name>FNF</ns1:name>
                <ns1:id>1</ns1:id>
            </ns0:site>
            <ns0:site>
                <ns1:code>A0001-0004</ns1:code>
                <ns1:dateLastVerified>2014-02-21</ns1:dateLastVerified>
                <ns1:name>Food</ns1:name>
                <ns1:id>3</ns1:id>
            </ns0:site>
            <ns0:site>
                <ns1:code>A0001-0003</ns1:code>
                <ns1:dateLastVerified>2014-02-21</ns1:dateLastVerified>
                <ns1:name>Produce</ns1:name>
                <ns1:id>2</ns1:id>
            </ns0:site>
            <ns0:siteContact>true</ns0:siteContact>
            <ns0:siteContactRole>
                <ns1:code>MANAGING DIRECTOR CHIEF EXECUTIVE</ns1:code>
                <ns1:isMandatorySiteRole>true</ns1:isMandatorySiteRole>
                <ns1:isMandatorySupplierRole>false</ns1:isMandatorySupplierRole>
                <ns1:isSiteRole>true</ns1:isSiteRole>
                <ns1:isSupplierRole>true</ns1:isSupplierRole>
                <ns1:localeData>
                    <ns0:description><![CDATA[Managing Director or Chief
Executive]]></ns0:description>
                    <ns0:id>88</ns0:id>
                </ns1:localeData>
                <ns1:showAsSpecContact>false</ns1:showAsSpecContact>
                <ns1:id>15</ns1:id>
            </ns0:siteContactRole>
            <ns0:siteContactRole>
```

```
            <ns1:code>HEAD TECHNICAL FUNCTION</ns1:code>
            <ns1:isMandatorySiteRole>false</ns1:isMandatorySiteRole>
            <ns1:isMandatorySupplierRole>false</ns1:isMandatorySupplierRole>
            <ns1:isSiteRole>true</ns1:isSiteRole>
            <ns1:isSupplierRole>true</ns1:isSupplierRole>
            <ns1:localeData>
                <ns0:description><![CDATA[Head of technical
function]]></ns0:description>
                <ns0:id>30</ns0:id>
            </ns1:localeData>
            <ns1:showAsSpecContact>false</ns1:showAsSpecContact>
            <ns1:id>5</ns1:id>
        </ns0:siteContactRole>
        <ns0:siteSelection>ALL_SITES</ns0:siteSelection>
        <ns0:supplierContact>true</ns0:supplierContact>
        <ns0:supplierContactRole>
            <ns1:code>EMERGENCY CONTACT</ns1:code>
            <ns1:isMandatorySiteRole>true</ns1:isMandatorySiteRole>
            <ns1:isMandatorySupplierRole>true</ns1:isMandatorySupplierRole>
            <ns1:isSiteRole>true</ns1:isSiteRole>
            <ns1:isSupplierRole>true</ns1:isSupplierRole>
            <ns1:localeData>
                <ns0:description><![CDATA[Emergency Contact]]></ns0:description>
                <ns0:id>93</ns0:id>
            </ns1:localeData>
            <ns1:showAsSpecContact>false</ns1:showAsSpecContact>
            <ns1:id>16</ns1:id>
        </ns0:supplierContactRole>
        <ns0:supplierContactRole>
            <ns1:code>MAIN CONTACT</ns1:code>
            <ns1:isMandatorySiteRole>true</ns1:isMandatorySiteRole>
            <ns1:isMandatorySupplierRole>true</ns1:isMandatorySupplierRole>
            <ns1:isSiteRole>true</ns1:isSiteRole>
            <ns1:isSupplierRole>true</ns1:isSupplierRole>
            <ns1:localeData>
                <ns0:description><![CDATA[Main Contact]]></ns0:description>
                <ns0:id>2</ns0:id>
            </ns1:localeData>
            <ns1:showAsSpecContact>false</ns1:showAsSpecContact>
            <ns1:id>1</ns1:id>
        </ns0:supplierContactRole>
        <ns0:udfData>
            <ns0:id>83</ns0:id>
        </ns0:udfData>
        <ns0:id>1</ns0:id>
    </ns0:supplierContacts>
    <ns0:supplierType>
        <ns1:code>MANUFACTURER</ns1:code>
        <ns1:localeData>
            <ns0:description><![CDATA[Manufacturer]]></ns0:description>
            <ns0:id>9</ns0:id>
        </ns1:localeData>
        <ns1:id>2</ns1:id>
    </ns0:supplierType>
    <ns0:udfData>
        <ns0:id>77</ns0:id>
    </ns0:udfData>
    <ns0:vatNumber>545274</ns0:vatNumber>
    <ns0:id>2</ns0:id>
</ns0:supplierFullDTO>
```

Error Responses

In the event that an error occurs, an HTTP 500 response is sent.

Check Record Modification Timestamp

Description

Retrieves the last modification time for a supplier.

Available from Version

1.9

Endpoint Address

/services/rest/supplier

HTTP Method

HEAD

Request Parameters

There are no request parameters, but the URL contains the {id} parameter that determines the record to load.

Response

If successful, an HTTP 200 response is sent containing the Last-Modified header to show the last modification data for that Supplier. For example:

```
HTTP/1.1 200 OK
Content-Length: 0
Expires: Thu, 01-Jan-1970 00:00:00 GMT
Set-Cookie: JSESSIONID=154osm4djwouv;Path=/creations-salesdemo
Date: Thu, 30 Apr 2015 10:33:05 GMT
Last-Modified: Fri, 28 Feb 2014 16:27:06 GMT
Server: Jetty(7.0.2.v20100331)
```

Error Responses

In the event that an error occurs, an HTTP 500 response is sent.

Create Record

Description

Creates a new supplier.

Available from Version

1.9

Endpoint Address

/services/rest/supplier

HTTP Method

POST

Request Body

The body of the request contains a SupplierFullDTO to specify the user to create. For example:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns1:supplierFullDTO
    xmlns:ns2="http://www.micros.com/creations/core/domain/dto/v1p0/simple"
    xmlns:ns1="http://www.micros.com/creations/core/domain/dto/v1p0/full">
    <ns1:code>WS1002</ns1:code>
    <ns1:contactDetails>
    </ns1:contactDetails>
    <ns1:email>joe.supplier1@example.com</ns1:email>
    <ns1:isActive>true</ns1:isActive>
    <ns1:mainAddress>
        </ns1:mainAddress>
    <ns1:name>Web Service Example Supplier Minimal</ns1:name>
    <ns1:status/>
    <ns1:supplierCodeConfirmed>true</ns1:supplierCodeConfirmed>
    <ns1:supplierContactName>Joe Bloggs</ns1:supplierContactName>
</ns1:supplierFullDTO>
```

Compared to the response for retrieving a supplier (which uses the same SupplierFullDTO type), this request is much shorter. This is achieved by only including the information relevant to creating the supplier. Where the record is linked to another record, such as the role, only the business keys (the code in this case) that allow that linked record to be identified are supplied. In general, this is the code attribute of the linked record, but in some cases, it is the status or a combination of values.

Response

If successful, an HTTP 200 response is sent with a body containing a SupplierLink:

```
<SupplierLink>
   <recordId>158</recordId>
   <recordLink>
      http://localhost:9090/creations-salesdemo/services/rest/supplier/158
   </recordLink>
   <code>WS1002</code>
   <name>Web Service Example Supplier Minimal</name>
</supplierLink>
```

The code of the request is mapped to the loginId of the response and the person/name is mapped to the name. The recordId is the ID of the newly created supplier and the recordLink is a URI to the supplier record that was created for use in a GET request.

Error Responses

If the supplied data does not result in a valid Supplier (such as the code missing), an HTTP 417 response is sent with a body message stating the validation errors. The request should not be reattempted with the same content.

If an internal error is experienced, an HTTP 500 response is sent.

Update Record

Description

Updates an existing Supplier.

Available from Version
1.9

Endpoint Address
/services/rest/supplier/{id}

HTTP Method
PUT

Request Body
The body of the request contains a SupplierUpdateDTO to specify how the supplier should appear after the update:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns3:supplierUpdateDTO
    xmlns:ns2="http://www.micros.com/creations/core/domain/dto/v1p0/simple"
    xmlns:ns1="http://www.micros.com/creations/core/domain/dto/v1p0/full"
    xmlns:ns3="http://www.micros.com/creations/core/domain/dto/v1p0/update">
    <ns1:code>WS0010</ns1:code>
    <ns1:contactDetails>
        </ns1:contactDetails>
    <ns1:email>joe.supplier1-upd@example.com</ns1:email>
    <ns1:isActive>false</ns1:isActive>
    <ns1:mainAddress>
        </ns1:mainAddress>
    <ns1:name>Web Service Example Supplier-upd</ns1:name>
    <ns1:status>POTENTIAL</ns1:status>
    <ns1:supplierCodeConfirmed>false</ns1:supplierCodeConfirmed>
    <ns1:supplierContactName>Joe Bloggs-upd</ns1:supplierContactName>
</ns3:supplierUpdateDTO>
```

The request content is similar to that for creating a Supplier, but crucially, the links to other top-level records (Sites, SiteContact, SupplierContact) are omitted. The omission of those ensures that when updating a supplier, you only have to specify the supplier details, and not the details for all the related records that you may not want to update and, in any case, should be updated to calls to their respective services. After the call, the supplier is updated to match the request. If a value is omitted from the request, the value is un-set on the supplier (except for the Sites, SiteContacts, and SupplierContacts).

Response
If successful, an HTTP 200 response is sent with a body containing a SupplierLink:

```
<SupplierLink>
   <recordId>158</recordId>
   <recordLink>
      http://localhost:9090/creations-salesdemo/services/supplier/user/158
   </recordLink>
   <code>WS1002</code>
   <name>Web Service Example Supplier Minimal</name>
</SupplierLink>
```

The code of the request is mapped to the loginId of the response and the person/name is mapped to the name. The recordId is the ID of the newly created supplier and the recordLink is a URI to the supplier record that was created for use in a GET request.

Error Responses

If the supplied data does not result in a valid Supplier (such as the code missing), an HTTP 417 response is sent with a body message stating the validation errors. The request should not be reattempted with the same content.

If an internal error occurs, an HTTP 500 response is sent.

SiteRestService

This section describes the API for managing sites.

List of Values

Description

Retrieves a list of sites in a paged list.

Available from Version

1.9

Endpoint Address

/services/rest/site

HTTP Method

GET

Request Parameters

Parameters are passed as URI parameters.

URI Parameters

Parameter Name	Mandatory/Optional	Value Type	Example
offset	Optional	Numeric	60
pageSize	Optional	Numeric	30

The offset and pageSize parameters control the paging of the list. The offset defines the first record to retrieve (0-based) and the pageSize defines how many records to include in the list.

Response

For a successful response, XML is returned with a SiteLinkList root element containing an entries element for each matched site with the recordId, code, name, and a recordLink URI to the SiteRestService Retrieve record service for this site. For example:

```
<SiteLinkList
    xmlns:ns1="http://www.micros.com/creations/core/domain/dto/v1p0/simple"
    xmlns:ns0="http://www.micros.com/creations/core/domain/dto/v1p0/full">
    <entries xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:type="siteLink">
        <recordId>97</recordId>
        <recordLink>
            http://localhost:9090/creations-salesdemo/services/rest/site/97
        </recordLink>
```

```
            <code>A0001-0001</code>
            <localName>Site 1 Local</localName>
            <name>Site 1</name>
            <supplierLink>
                <recordId>95</recordId>
                <recordLink>
                   http://localhost:9090/creations-salesdemo/services/rest/supplier/95
                </recordLink>
                <code>A0001</code>
                <localName>Trading Local name</localName>
                <name>Trading</name>
            </supplierLink>
        </entries>
        <totalRecords>1</totalRecords>
        <entryArray>
            <recordId>97</recordId>
            <recordLink>
                http://localhost:9090/creations-salesdemo/services/rest/site/97
            </recordLink>
            <code>A0001-0001</code>
            <localName>Site 1 Local</localName>
            <name>Site 1</name>
            <supplierLink>
                <recordId>95</recordId>
                <recordLink>
                   http://localhost:9090/creations-salesdemo/services/rest/supplier/95
                </recordLink>
                <code>A0001</code>
                <localName>Trading Local name</localName>
                <name>Trading</name>
            </supplierLink>
        </entryArray>
    </SiteLinkList>
```

Error Responses

In the event that an error occurs, an HTTP 500 response is sent.

Retrieve Record

Description

Retrieves a single site's details.

Available from Version

1.9

Endpoint Address

/services/rest/site/{id}

HTTP Method

GET

Request Parameters

There are no request parameters, but the URL contains the {id} parameter that determines the record to load.

Response

For a successful response, XML is returned with a siteFullDTO root element containing the details of the site requested. For example:

```
<ns0:siteFullDTO
    xmlns:ns1="http://www.micros.com/creations/core/domain/dto/v1p0/simple"
    xmlns:ns0="http://www.micros.com/creations/core/domain/dto/v1p0/full">
    <ns0:allTMs>
        <ns1:code>producttechnologist</ns1:code>
        <ns1:disabled>false</ns1:disabled>
        <ns1:externalAuthenticationUser>false</ns1:externalAuthenticationUser>
        <ns1:timeZone>Europe/London</ns1:timeZone>
        <ns1:userType>RETAILER</ns1:userType>
        <ns1:id>145</ns1:id>
    </ns0:allTMs>
    <ns0:billingAddress>
        <ns0:id>844</ns0:id>
    </ns0:billingAddress>
    <ns0:businessCategories>
        <ns1:code>CATEGORY1</ns1:code>
        <ns1:localeData>
            <ns0:description><![CDATA[Category Level 1]]></ns0:description>
            <ns0:path><![CDATA[Category Level 1]]></ns0:path>
            <ns0:id>44</ns0:id>
        </ns1:localeData>
        <ns1:specificationType>BWS</ns1:specificationType>
        <ns1:specificationType>CNF</ns1:specificationType>
        <ns1:specificationType>PRODUCE</ns1:specificationType>
        <ns1:specificationType>FOOD</ns1:specificationType>
        <ns1:specificationType>FNF</ns1:specificationType>
        <ns1:topLevelCategory>true</ns1:topLevelCategory>
        <ns1:id>34</ns1:id>
    </ns0:businessCategories>
    <ns0:businessCategories>
        <ns1:code>CATEGORY2A</ns1:code>
        <ns1:localeData>
            <ns0:description><![CDATA[Category Level 2a]]></ns0:description>
            <ns0:path><![CDATA[Category Level 1 / Category Level 2a]]></ns0:path>
            <ns0:id>38</ns0:id>
        </ns1:localeData>
        <ns1:specificationType>BWS</ns1:specificationType>
        <ns1:specificationType>CNF</ns1:specificationType>
        <ns1:specificationType>PRODUCE</ns1:specificationType>
        <ns1:specificationType>FOOD</ns1:specificationType>
        <ns1:specificationType>FNF</ns1:specificationType>
        <ns1:id>35</ns1:id>
    </ns0:businessCategories>
    <ns0:code>A0001-0001</ns0:code>
    <ns0:conductedBy>Conducted By test data</ns0:conductedBy>
    <ns0:contactDetails>
        <ns0:phoneNumber>0123654987</ns0:phoneNumber>
        <ns0:id>646</ns0:id>
    </ns0:contactDetails>
    <ns0:leadBusinessCategory>
        <ns1:code>CATEGORY1</ns1:code>
        <ns1:localeData>
            <ns0:description><![CDATA[Category Level 1]]></ns0:description>
            <ns0:path><![CDATA[Category Level 1]]></ns0:path>
            <ns0:id>44</ns0:id>
        </ns1:localeData>
```

```
    <ns1:specificationType>BWS</ns1:specificationType>
    <ns1:specificationType>CNF</ns1:specificationType>
    <ns1:specificationType>PRODUCE</ns1:specificationType>
    <ns1:specificationType>FOOD</ns1:specificationType>
    <ns1:specificationType>FNF</ns1:specificationType>
    <ns1:topLevelCategory>true</ns1:topLevelCategory>
    <ns1:id>34</ns1:id>
</ns0:leadBusinessCategory>
<ns0:leadTechnicalManager>
    <ns1:code>producttechnologist</ns1:code>
    <ns1:disabled>false</ns1:disabled>
    <ns1:externalAuthenticationUser>false</ns1:externalAuthenticationUser>
    <ns1:timeZone>Europe/London</ns1:timeZone>
    <ns1:userType>RETAILER</ns1:userType>
    <ns1:id>145</ns1:id>
</ns0:leadTechnicalManager>
<ns0:localName>Site 1 Local</ns0:localName>
<ns0:mainAddress>
    <ns0:country>
    <ns1:code>GB</ns1:code>
    <ns1:countryType>COUNTRY</ns1:countryType>
    <ns1:internetDomainSuffix>co.uk</ns1:internetDomainSuffix>
    <ns1:localeData>
        <ns0:description><![CDATA[United Kingdom]]></ns0:description>
        <ns0:locale>en_US</ns0:locale>
        <ns0:id>15034</ns0:id>
    </ns1:localeData>
    <ns1:localeData>
        <ns0:description><![CDATA[United Kingdom]]></ns0:description>
            <ns0:id>15045</ns0:id>
         </ns1:localeData>
         <ns1:id>1013</ns1:id>
    </ns0:country>
    <ns0:line1>Site Address 1 test</ns0:line1>
    <ns0:line2>Site Address 2 test</ns0:line2>
    <ns0:line3>Site Address 3 test</ns0:line3>
    <ns0:line4>Site Address 4 test</ns0:line4>
    <ns0:postalCode>Post Code</ns0:postalCode>
    <ns0:id>845</ns0:id>
</ns0:mainAddress>
<ns0:name>Site 1</ns0:name>
<ns0:productionDetail>
    <ns0:id>97</ns0:id>
</ns0:productionDetail>
<ns0:reference>
    <ns0:referenceCertNo>987654321</ns0:referenceCertNo>
    <ns0:referenceType>
        <ns1:code>EC_LICENCE</ns1:code>
        <ns1:localeData>
            <ns0:description><![CDATA[EC Licence Number]]></ns0:description>
            <ns0:id>41</ns0:id>
        </ns1:localeData>
        <ns1:id>41</ns1:id>
    </ns0:referenceType>
    <ns0:id>4</ns0:id>
</ns0:reference>
<ns0:riskLevel>
    <ns1:code>HIGH_RISK</ns1:code>
    <ns1:localeData>
        <ns0:description><![CDATA[High Risk]]></ns0:description>
```

```
            <ns0:id>1896</ns0:id>
        </ns1:localeData>
        <ns1:id>128</ns1:id>
    </ns0:riskLevel>
    <ns0:siteTopGrade>
        <ns1:code>A</ns1:code>
        <ns1:localeData>
            <ns0:description><![CDATA[A]]></ns0:description>
            <ns0:id>120</ns0:id>
        </ns1:localeData>
        <ns1:id>10</ns1:id>
    </ns0:siteTopGrade>
    <ns0:siteType>
        <ns1:code>SITE_TYPE_EXAMPLE</ns1:code>
        <ns1:localeData>
            <ns0:description><![CDATA[Site Type]]></ns0:description>
            <ns0:id>4</ns0:id>
        </ns1:localeData>
        <ns1:id>4</ns1:id>
    </ns0:siteType>
    <ns0:siteStatus>
        <ns1:approvedStatus>false</ns1:approvedStatus>
        <ns1:createInitialAudit>false</ns1:createInitialAudit>
        <ns1:localeData>
            <ns0:description><![CDATA[Awaiting Registration]]></ns0:description>
            <ns0:id>485</ns0:id>
        </ns1:localeData>
        <ns1:localeData>
            <ns0:description><![CDATA[Oczekuje na rejestracj?]]></ns0:description>
            <ns0:locale>pl_PL</ns0:locale>
            <ns0:id>472</ns0:id>
        </ns1:localeData>
        <ns1:status>AWAITING REGISTRATION</ns1:status>
        <ns1:id>19</ns1:id>
    </ns0:siteStatus>
    <ns0:supplier>
        <ns1:code>A0001</ns1:code>
        <ns1:email>black.hole@example.com</ns1:email>
        <ns1:isActive>false</ns1:isActive>
        <ns1:localName>Trading Local name</ns1:localName>
        <ns1:name>Trading</ns1:name>
        <ns1:status>AWAITING REGISTRATION</ns1:status>
        <ns1:supplierCodeConfirmed>false</ns1:supplierCodeConfirmed>
        <ns1:supplierContactName>Andrew Example</ns1:supplierContactName>
        <ns1:id>95</ns1:id>
    </ns0:supplier>
    <ns0:udfData>
        <ns0:id>567</ns0:id>
    </ns0:udfData>
    <ns0:id>97</ns0:id>
</ns0:siteFullDTO>
```

Error Responses

In the event that an error occurs, an HTTP 500 response is sent.

Check Record Modification Timestamp

Description
Retrieves the last modification time for a site.

Available from Version
1.9

Endpoint Address
/services/rest/site/{id}

HTTP Method
HEAD

Request Parameters
There are no request parameters, but the URL contains the {id} parameter that determines the record to load.

Response
If successful, an HTTP 200 response is sent containing the Last-Modified header to show the last modification data for that Site. For example:

```
HTTP/1.1 200 OK
Content-Length: 0
Expires: Thu, 01-Jan-1970 00:00:00 GMT
Set-Cookie: JSESSIONID=154osm4djwouv;Path=/creations-salesdemo
Date: Thu, 30 Apr 2015 10:33:05 GMT
Last-Modified: Fri, 28 Feb 2014 16:27:06 GMT
Server: Jetty(7.0.2.v20100331)
```

Error Responses
In the event that an error occurs, an HTTP 500 response is sent.

Create Record

Description
Creates a new site.

Available from Version
1.9

Endpoint Address
/services/rest/site

HTTP Method
POST

Request Body
The body of the request contains a SiteFullDTO to specify the site to create. For example:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

```
<ns1:siteFullDTO
    xmlns:ns2="http://www.micros.com/creations/core/domain/dto/v1p0/simple"
    xmlns:ns1="http://www.micros.com/creations/core/domain/dto/v1p0/full">
    <ns1:code>A0001-1001</ns1:code>
    <ns1:name>Web Service Example Site</ns1:name>
    <ns1:siteStatus>
        <ns2:status>ACTIVE</ns2:status>
    </ns1:siteStatus>
    <ns1:supplier>
    <ns2:code>A0001</ns2:code>
    </ns1:supplier>
</ns1:siteFullDTO>
```

Compared to the response for retrieving a site (which uses the same SiteFullDTO type), this request is much shorter. This is achieved by only including the information relevant to creating the site. Where the record is linked to another record, such as the role, only the business keys (the code in this case) that allow that linked record to be identified are supplied. In general, this is the code attribute of the linked record, but in some cases, it is the status or a combination of values.

Response

If successful, an HTTP 200 response is sent with a body containing a SiteLink:

```
<SiteLink>
   <recordId>97</recordId>
   <recordLink>
      http://localhost:9090/creations-salesdemo/services/rest/site/97
   </recordLink>
   <code>A0001-0001</code>
   <localName></localName>
   <name>Web Service Example Site</name>
   <supplierLink>
      <recordId>95</recordId>
      <recordLink>
         http://localhost:9090/creations-salesdemo/services/rest/supplier/95
      </recordLink>
      <code>A0001</code>
      <localName>Trading Local name</localName>
      <name>Trading</name>
   </supplierLink>
</SiteLink>
```

The recordId is the ID of the newly created site and the recordLink is a URI to the site record that was created for use in a GET request.

Error Responses

If the supplied data does not result in a valid Site (such as the code missing), an HTTP 417 response is sent with a body message stating the validation errors. The request should not be reattempted with the same content.

If an internal error is experienced, an HTTP 500 response is sent.

Update Record

Description

Updates an existing Site.

Available from Version
1.9

Endpoint Address
/services/rest/site/{id}

HTTP Method
PUT

Request Body
The body of the request contains a SiteUpdateDTO to specify how the site should appear after the update:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns3:siteUpdateDTO
    xmlns:ns2="http://www.micros.com/creations/core/domain/dto/v1p0/simple"
    xmlns:ns1="http://www.micros.com/creations/core/domain/dto/v1p0/full"
    xmlns:ns3="http://www.micros.com/creations/core/domain/dto/v1p0/update">
    <ns1:code>A0001-1100</ns1:code>
    <ns1:name>Web Service Example Site-upd</ns1:name>
    <ns1:siteStatus>
        <ns2:status>DELISTED</ns2:status>
    </ns1:siteStatus>
    <ns1:supplier>
        <ns2:code>A0001</ns2:code>
    </ns1:supplier>
</ns3:siteUpdateDTO>
```

The request content is similar to that for creating a site, but crucially, the links to other top-level records (SiteContact) are omitted. The omission of those ensures that when updating a suite, you only have to specify the site details, and not the details for all the related records that you may not want to update and, in any case, should be updated to calls to their respective services. After the call, the site is updated to match the request. If a value is omitted from the request, the value is un-set on the site (excepting the SiteContacts).

Response
If successful, an HTTP 200 response is sent with a body containing a SiteLink:

```
<SiteLink>
   <recordId>97</recordId>
   <recordLink>
      http://localhost:9090/creations-salesdemo/services/rest/site/97
   </recordLink>
   <code>A0001-1100</code>
   <localName></localName>
   <name>Web Service Example Site-upd</name>
   <supplierLink>
      <recordId>95</recordId>
      <recordLink>
         http://localhost:9090/creations-salesdemo/services/rest/supplier/95
      </recordLink>
      <code>A0001</code>
      <localName>Trading Local name</localName>
      <name>Trading</name>
   </supplierLink>
</SiteLink>
```

The recordId is the ID of the newly created site and the recordLink is a URI to the site record that was created for use in a GET request.

Error Responses

If the supplied data does not result in a valid Site (such as the code missing), an HTTP 417 response is sent with a body message stating the validation errors. The request should not be reattempted with the same content.

If an internal error occurs, an HTTP 500 response is sent.

ContactRestService

This section describes the API for managing supplier and site contacts.

List of Values

Description

Retrieves a list of contacts in a paged list.

Available from Version

1.9

Endpoint Address

/services/rest/contact

HTTP Method

GET

Request Parameters

Parameters are passed as URI parameters.

URI Parameters

Parameter Name	Mandatory/Optional	Value Type	Example
offset	Optional	Numeric	60
pageSize	Optional	Numeric	30

The offset and pageSize parameters control the paging of the list. The offset defines the first record to retrieve (0-based) and the pageSize defines how many records to include in the list.

Response

For a successful response, XML is returned with a ContactLinkList root element containing an entries element for each matched contact with the recordId, email, name, siteContact, supplierContact flags and a recordLink URI to the ContactRestService Retrieve record service for this contact. For example:

```
<ContactLinkList
    xmlns:ns1="http://www.micros.com/creations/core/domain/dto/v1p0/simple"
    xmlns:ns0="http://www.micros.com/creations/core/domain/dto/v1p0/full">
    <entries xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
        xsi:type="contactLink">
        <recordId>91</recordId>
        <recordLink>
            http://localhost:9090/creations-salesdemo/services/rest/contact/91
        </recordLink>
        <email>black.hole@example.com</email>
        <name>Andrew Example</name>
        <siteContact>false</siteContact>
        <supplierContact>true</supplierContact>
    </entries>
    <totalRecords>1</totalRecords>
    <entryArray>
        <recordId>91</recordId>
        <recordLink>
            http://localhost:9090/creations-salesdemo/services/rest/contact/91
        </recordLink>
        <email>black.hole@example.com</email>
        <name>Andrew Example</name>
        <siteContact>false</siteContact>
        <supplierContact>true</supplierContact>
    </entryArray>
</ContactLinkList>
```

Error Responses

In the event that an error occurs, an HTTP 500 response is sent.

Retrieve Record

Description

Retrieves a single contact's details.

Available from Version

1.10

Endpoint Address

/services/rest/contact/{id}

HTTP Method

GET

Request Parameters

There are no request parameters, but the URL contains the {id} parameter that determines the record to load.

Response

For a successful response, XML is returned with a contactAndPersonDTO root element containing the details of the contact requested and the person it associates to. For example:

```
<ContactAndPersonDTO
   xmlns:ns1="http://www.micros.com/creations/core/domain/dto/v1p0/simple"
    xmlns:ns0="http://www.micros.com/creations/core/domain/dto/v1p0/full">
    <contactFullDTO>
        <ns0:address>
             <ns0:id>843</ns0:id>
```

```
        </ns0:address>
        <ns0:contactDetails>
            <ns0:phoneNumber>0132465798</ns0:phoneNumber>
            <ns0:id>645</ns0:id>
        </ns0:contactDetails>
        <ns0:dtype>SupplierContact</ns0:dtype>
        <ns0:person>
            <ns1:email>black.hole@example.com</ns1:email>
            <ns1:name>Andrew Example</ns1:name>
            <ns1:id>159</ns1:id>
        </ns0:person>
        <ns0:siteContact>false</ns0:siteContact>
        <ns0:siteSelection>SELECTED_SITES</ns0:siteSelection>
        <ns0:supplierContact>true</ns0:supplierContact>
        <ns0:supplierContactRole>
            <ns1:code>MAIN CONTACT</ns1:code>
            <ns1:isMandatorySiteRole>true</ns1:isMandatorySiteRole>
            <ns1:isMandatorySupplierRole>true</ns1:isMandatorySupplierRole>
            <ns1:isSiteRole>true</ns1:isSiteRole>
            <ns1:isSupplierRole>true</ns1:isSupplierRole>
            <ns1:localeData>
                <ns0:description><![CDATA[Main Contact]]></ns0:description>
                <ns0:id>308</ns0:id>
            </ns1:localeData>
            <ns1:showAsSpecContact>false</ns1:showAsSpecContact>
            <ns1:id>52</ns1:id>
        </ns0:supplierContactRole>
        <ns0:udfData>
            <ns0:id>566</ns0:id>
        </ns0:udfData>
        <ns0:id>91</ns0:id>
        <ns0:company>
            <ns1:code>A0001</ns1:code>
            <ns1:email>black.hole@example.com</ns1:email>
            <ns1:isActive>false</ns1:isActive>
            <ns1:localName>Trading Local name</ns1:localName>
            <ns1:name>Trading</ns1:name>
            <ns1:status>AWAITING REGISTRATION</ns1:status>
            <ns1:supplierCodeConfirmed>false</ns1:supplierCodeConfirmed>
            <ns1:supplierContactName>Andrew Example</ns1:supplierContactName>
            <ns1:id>95</ns1:id>
        </ns0:company>
    </contactFullDTO>
    <personFullDTO>
        <ns0:address>
            <ns0:id>842</ns0:id>
        </ns0:address>
        <ns0:contactDetails>
            <ns0:id>644</ns0:id>
        </ns0:contactDetails>
        <ns0:email>black.hole@example.com</ns0:email>
        <ns0:language>
            <ns1:code>en_GB</ns1:code>
            <ns1:isActive>true</ns1:isActive>
            <ns1:localeData>
                <ns0:description><![CDATA[English (British)]]></ns0:description>
                <ns0:id>116</ns0:id>
            </ns1:localeData>
            <ns1:localeData>
                <ns0:description><![CDATA[Anglais
```

```
(Britannique)]]></ns0:description>
                <ns0:locale>fr</ns0:locale>
                <ns0:id>112</ns0:id>
            </ns1:localeData>
            <ns1:id>22</ns1:id>
        </ns0:language>
        <ns0:name>Andrew Example</ns0:name>
        <ns0:supplier>
            <ns1:code>A0001</ns1:code>
            <ns1:email>black.hole@example.com</ns1:email>
            <ns1:isActive>false</ns1:isActive>
            <ns1:localName>Trading Local name</ns1:localName>
            <ns1:name>Trading</ns1:name>
            <ns1:status>AWAITING REGISTRATION</ns1:status>
            <ns1:supplierCodeConfirmed>false</ns1:supplierCodeConfirmed>
            <ns1:supplierContactName>Andrew Example</ns1:supplierContactName>
            <ns1:id>95</ns1:id>
        </ns0:supplier>
        <ns0:id>159</ns0:id>
    </personFullDTO>
</ContactAndPersonDTO>
```

Error Responses

In the event that an error occurs, an HTTP 500 response is sent.

Check Record Modification Timestamp

Description

Retrieves the last modification time for a contact.

Available from Version

1.10

Endpoint Address

/services/rest/contact/{id}

HTTP Method

HEAD

Request Parameters

There are no request parameters, but the URL contains the {id} parameter that determines the record to load.

Response

If successful, an HTTP 200 response is sent containing the Last-Modified header to show the last modification data for that Contact. For example:

```
HTTP/1.1 200 OK
Content-Length: 0
Expires: Thu, 01-Jan-1970 00:00:00 GMT
Set-Cookie: JSESSIONID=154osm4djwouv;Path=/creations-salesdemo
Date: Thu, 30 Apr 2015 10:33:05 GMT
Last-Modified: Fri, 28 Feb 2014 16:27:06 GMT
Server: Jetty(7.0.2.v20100331)
```

Error Responses

In the event that an error occurs, an HTTP 500 response is sent.

Create Record

Description

Creates a new Contact.

Available from Version

1.10

Endpoint Address

/services/rest/contact

HTTP Method

POST

Request Body

The body of the request contains a ContactAndPersonDTO to specify the contact (and associated person) to create. For example:

```
<ContactAndPersonDTO
    xmlns:ns1="http://www.micros.com/creations/core/domain/dto/v1p0/simple"
    xmlns:ns0="http://www.micros.com/creations/core/domain/dto/v1p0/full">
    <contactFullDTO>
        <ns0:contactDetails>
            <ns0:phoneNumber>0132465798</ns0:phoneNumber>
        </ns0:contactDetails>
        <ns0:dtype>SupplierContact</ns0:dtype>
        <ns0:person>
            <ns1:email>black.hole@example.com</ns1:email>
            <ns1:name>Andrew Example</ns1:name>
        </ns0:person>
        <ns0:siteContact>false</ns0:siteContact>
        <ns0:siteSelection>SELECTED_SITES</ns0:siteSelection>
        <ns0:supplierContact>true</ns0:supplierContact>
        <ns0:supplierContactRole>
            <ns1:code>MAIN CONTACT</ns1:code>
        </ns0:supplierContactRole>
        <ns0:company>
            <ns1:code>A0001</ns1:code>
        </ns0:company>
    </contactFullDTO>
    <personFullDTO>
        <ns0:email>black.hole@example.com</ns0:email>
        <ns0:language>
            <ns1:code>en_GB</ns1:code>
        </ns0:language>
        <ns0:name>Andrew Example</ns0:name>
        <ns0:supplier>
            <ns1:code>A0001</ns1:code>
        </ns0:supplier>
    </personFullDTO>
</ContactAndPersonDTO>
```

Compared to the response for retrieving a contact (which uses the same ContactAndPersonDTO type), this request is much shorter. This is achieved by only including the information relevant to creating the contact. Where the record is linked to another record, such as the role, only the business keys (the code in this case) that allow that linked record to be identified are supplied. In general, this is the code attribute of the linked record, but in some cases, it is the status or a combination of values.

Response

If successful, an HTTP 200 response is sent with a body containing a ContactLink:

```
<ContactLink>
   <recordId>91</recordId>
   <recordLink>
      http://localhost:9090/creations-salesdemo/services/rest/contact/91
   </recordLink>
   <email>black.hole@example.com</email>
   <name>Andrew Example</name>
   <siteContact>false</siteContact>
   <supplierContact>true</supplierContact>
</ContactLink>
```

The recordId is the ID of the newly created contact and the recordLink is a URI to the contact record that was created for use in a GET request.

Error Responses

If the supplied data does not result in a valid Contact (such as the supplier's code missing), an HTTP 417 response is sent with a body message stating the validation errors. The request should not be reattempted with the same content.

If an internal error is experienced, an HTTP 500 response is sent.

Update Record

Description

Updates an existing Contact.

Available from Version

1.10

Endpoint Address

/services/rest/contact/{id}

HTTP Method

PUT

Request Body

The body of the request contains a ContactAndPersonDTO to specify how the contact should appear after the update:

```
<ContactAndPersonDTO
    xmlns:ns1="http://www.micros.com/creations/core/domain/dto/v1p0/simple"
    xmlns:ns0="http://www.micros.com/creations/core/domain/dto/v1p0/full">
    <contactFullDTO>
        <ns0:contactDetails>
```

```
            <ns0:phoneNumber>0132465798-2</ns0:phoneNumber>
        </ns0:contactDetails>
        <ns0:dtype>SupplierContact</ns0:dtype>
        <ns0:person>
            <ns1:email>black.hole-2@example.com</ns1:email>
            <ns1:name>Andrew Example-2</ns1:name>
        </ns0:person>
        <ns0:siteContact>false</ns0:siteContact>
        <ns0:siteSelection>SELECTED_SITES</ns0:siteSelection>
        <ns0:supplierContact>true</ns0:supplierContact>
        <ns0:supplierContactRole>
            <ns1:code>MAIN CONTACT</ns1:code>
        </ns0:supplierContactRole>
        <ns0:id>91</ns0:id>
        <ns0:company>
            <ns1:code>A0001</ns1:code>
        </ns0:company>
    </contactFullDTO>
    <personFullDTO>
        <ns0:email>black.hole-2@example.com</ns0:email>
        <ns0:language>
            <ns1:code>en_GB</ns1:code>
        </ns0:language>
        <ns0:name>Andrew Example-2</ns0:name>
        <ns0:supplier>
            <ns1:code>A0001</ns1:code>
        </ns0:supplier>
    </personFullDTO>
</ContactAndPersonDTO>
```

The request content is the same as that for creating a contact. After the call, the contact is updated to match the request. If a value is omitted from the request, the value is un-set on the contact.

Response

If successful, an HTTP 200 response is sent with a body containing a ContactLink:

```
<ContactLink>
   <recordId>91</recordId>

<recordLink>http://localhost:9090/creations-salesdemo/services/rest/contact/91</re
cordLink>
   <email>black.hole@example.com</email>
   <name>Andrew Example</name>
   <siteContact>false</siteContact>
   <supplierContact>true</supplierContact>
</ContactLink>
```

The recordId is the ID of the updated contact and the recordLink is a URI to the contact record that was created for use in a GET request.

Error Responses

If the supplied data does not result in a valid Contact (such as the supplier code missing), an HTTP 417 response is sent with a body message stating the validation errors. The request should not be reattempted with the same content.

If an internal error is experienced, an HTTP 500 response is sent.

ProductRecordRestService

This section describes the API for managing product records.

List of Values

Description

Retrieves a list of product records in a paged list.

Available from Version

1.9

Endpoint Address

/services/rest/productRecord

HTTP Method

GET

Request Parameters

Parameters are passed as URI parameters.

URI Parameters

Parameter Name	Mandatory/Optional	Value Type	Example
offset	Optional	Numeric	60
pageSize	Optional	Numeric	30

The offset and pageSize parameters control the paging of the list. The offset defines the first record to retrieve (0-based) and the pageSize defines how many records to include in the list.

Response

For a successful response, XML is returned with a ProductRecordLinkList root element containing an entries element for each matched product record with the recordId, code, title, and a recordLink URI to the ProductRecordRestService Retrieve record service for this product record. For example:

```
<ProductRecordLinkList>
    <entries xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:type="productRecordLink">
        <recordId>1</recordId>
        <recordLink>
           http://localhost:9090/creations-salesdemo/services/rest/productRecord/1
        </recordLink>
        <code>1</code>
        <title>Caesar Salad Kit</title>
    </entries>
    <nextPage>
http://localhost:9090/creations-salesdemo/services/rest/productRecord/?offset=1&am
p;pageSize=1
    </nextPage>
    <totalRecords>153</totalRecords>
    <entryArray>
```

```
        <recordId>1</recordId>
        <recordLink>
           http://localhost:9090/creations-salesdemo/services/rest/productRecord/1
        </recordLink>
        <code>1</code>
        <title>Caesar Salad Kit</title>
    </entryArray>
</ProductRecordLinkList>
```

Error Responses

In the event that an error occurs, an HTTP 500 response is sent.

Retrieve Record

Description

Retrieves a single product record's details.

Available from Version

1.9

Endpoint Address

/services/rest/productRecord/{id}

HTTP Method

GET

Request Parameters

There are no request parameters, but the URL contains the {id} parameter that determines the record to load.

Response

For a successful response, XML is returned with a productRecordFullDTO root element containing the details of the Product Record. For example:

```
<ns0:productRecordFullDTO
    xmlns:ns1="http://www.micros.com/creations/core/domain/dto/v1p0/simple"
    xmlns:ns0="http://www.micros.com/creations/core/domain/dto/v1p0/full">
    <ns0:addSuppInfo>
        <ns0:id>1</ns0:id>
    </ns0:addSuppInfo>
    <ns0:businessCategory>
        <ns1:code>CATEGORY2A</ns1:code>
        <ns1:localeData>
            <ns0:description><![CDATA[Salads]]></ns0:description>
            <ns0:path><![CDATA[Chilled / Salads]]></ns0:path>
            <ns0:id>5</ns0:id>
        </ns1:localeData>
        <ns1:specificationType>CNF</ns1:specificationType>
        <ns1:specificationType>PRODUCE</ns1:specificationType>
        <ns1:specificationType>FOOD</ns1:specificationType>
        <ns1:specificationType>FNF</ns1:specificationType>
        <ns1:id>2</ns1:id>
    </ns0:businessCategory>
    <ns0:code>1</ns0:code>
    <ns0:disciplineDetail>
```

```
        <ns0:testDiscipline>
            <ns1:code>CHEMICAL</ns1:code>
            <ns1:isActive>true</ns1:isActive>
            <ns1:localeData>
                <ns0:description><![CDATA[Chemical]]></ns0:description>
                <ns0:id>1</ns0:id>
            </ns1:localeData>
            <ns1:id>1</ns1:id>
        </ns0:testDiscipline>
        <ns0:id>1</ns0:id>
    </ns0:disciplineDetail>
    <ns0:disciplineDetail>
            <ns0:testDiscipline>
            <ns1:code>PHYSICAL</ns1:code>
            <ns1:isActive>true</ns1:isActive>
                <ns1:localeData>
                <ns0:description><![CDATA[Physical]]></ns0:description>
                <ns0:id>2</ns0:id>
            </ns1:localeData>
            <ns1:id>2</ns1:id>
        </ns0:testDiscipline>
        <ns0:id>2</ns0:id>
    </ns0:disciplineDetail>
    <ns0:disciplineDetail>
        <ns0:testDiscipline>
            <ns1:code>MICRO</ns1:code>
            <ns1:isActive>true</ns1:isActive>
            <ns1:localeData>
                <ns0:description><![CDATA[Microbiological]]></ns0:description>
                <ns0:id>3</ns0:id>
            </ns1:localeData>
            <ns1:id>3</ns1:id>
        </ns0:testDiscipline>
        <ns0:id>3</ns0:id>
    </ns0:disciplineDetail>
    <ns0:productCovered>
        <ns0:productTitle>Caesar Salad Kit</ns0:productTitle>
        <ns0:quantity>1</ns0:quantity>
        <ns0:retailerProductNumber>1</ns0:retailerProductNumber>
        <ns0:variantName><![CDATA[Caesar Salad Kit]]></ns0:variantName>
        <ns0:id>1</ns0:id>
    </ns0:productCovered>
    <ns0:productRecordOtherUser>
        <ns0:mandatoryUser>true</ns0:mandatoryUser>
        <ns0:multipleUser>false</ns0:multipleUser>
        <ns0:role>
            <ns1:code>BUYER</ns1:code>
            <ns1:localeData>
                <ns0:description><![CDATA[Buyer]]></ns0:description>
                <ns0:id>4</ns0:id>
            </ns1:localeData>
            <ns1:id>4</ns1:id>
        </ns0:role>
        <ns0:user>
            <ns1:code>Richard Admin</ns1:code>
            <ns1:disabled>false</ns1:disabled>
            <ns1:timeZone>Europe/London</ns1:timeZone>
            <ns1:userType>RETAILER</ns1:userType>
            <ns1:id>25</ns1:id>
        </ns0:user>
```

```
            <ns0:id>1</ns0:id>
        </ns0:productRecordOtherUser>
        <ns0:productRecordOtherUser>
            <ns0:mandatoryUser>true</ns0:mandatoryUser>
            <ns0:multipleUser>false</ns0:multipleUser>
            <ns0:role>
                <ns1:code>PRODUCT DEVELOPMENT MANAGER</ns1:code>
                <ns1:localeData>
                    <ns0:description><![CDATA[Product Development
Manager]]></ns0:description>
                    <ns0:id>3</ns0:id>
                </ns1:localeData>
                <ns1:id>3</ns1:id>
            </ns0:role>
            <ns0:user>
                <ns1:code>productdevelopmentmanager</ns1:code>
                <ns1:disabled>false</ns1:disabled>
                <ns1:timeZone>Europe/London</ns1:timeZone>
                <ns1:userType>RETAILER</ns1:userType>
                <ns1:id>6</ns1:id>
            </ns0:user>
            <ns0:id>2</ns0:id>
        </ns0:productRecordOtherUser>
        <ns0:projectName>Caesar Salad Kit</ns0:projectName>
        <ns0:site>
            <ns1:code>A0001-0004</ns1:code>
            <ns1:dateLastVerified>2014-02-21</ns1:dateLastVerified>
            <ns1:name>Food</ns1:name>
            <ns1:id>3</ns1:id>
        </ns0:site>
        <ns0:specTypeFormat>
            <ns1:code>FOODUK</ns1:code>
            <ns1:isActive>true</ns1:isActive>
            <ns1:localeData>
                <ns0:declaration><![CDATA[Food Declaration
section]]></ns0:declaration>
                <ns0:description><![CDATA[Food]]></ns0:description>
                <ns0:id>5</ns0:id>
            </ns1:localeData>
            <ns1:specType>FOOD</ns1:specType>
            <ns1:id>5</ns1:id>
        </ns0:specTypeFormat>
        <ns0:status>
            <ns1:changeDialogType>NONE</ns1:changeDialogType>
            <ns1:localeData>
                <ns0:description><![CDATA[??]]></ns0:description>
                <ns0:locale>zh_CN</ns0:locale>
                <ns0:id>36</ns0:id>
            </ns1:localeData>
            <ns1:localeData>
                <ns0:description><![CDATA[Active]]></ns0:description>
                <ns0:id>33</ns0:id>
            </ns1:localeData>
            <ns1:status>ACTIVE</ns1:status>
            <ns1:id>3</ns1:id>
        </ns0:status>
        <ns0:statusHistory>
            <ns0:localeData>
            <ns0:statusFrom>Draft</ns0:statusFrom>
            <ns0:statusTo>Active</ns0:statusTo>
```

```
            <ns0:id>263</ns0:id>
            </ns0:localeData>
            <ns0:id>1</ns0:id>
        </ns0:statusHistory>
        <ns0:supplier>
            <ns1:billingAddressTo>Amber Example</ns1:billingAddressTo>
            <ns1:billingContact>Amber Example</ns1:billingContact>
            <ns1:billingEmail>aexample@example.com</ns1:billingEmail>
            <ns1:billingFax>0123456798</ns1:billingFax>
            <ns1:billingPhone>0123654987</ns1:billingPhone>
            <ns1:code>A0001</ns1:code>
            <ns1:companyNumber>123981</ns1:companyNumber>
            <ns1:dateLastVerified>2014-02-21</ns1:dateLastVerified>
            <ns1:email>aexample@example.com</ns1:email>
            <ns1:isActive>true</ns1:isActive>
            <ns1:name>FOODS</ns1:name>
            <ns1:status>REGISTERED</ns1:status>
            <ns1:supplierCodeConfirmed>true</ns1:supplierCodeConfirmed>
            <ns1:supplierContactName>Amber Example</ns1:supplierContactName>
            <ns1:vatNumber>545274</ns1:vatNumber>
            <ns1:id>2</ns1:id>
        </ns0:supplier>
        <ns0:technologist>
            <ns1:code>techadmin</ns1:code>
            <ns1:disabled>false</ns1:disabled>
            <ns1:timeZone>Europe/London</ns1:timeZone>
            <ns1:userType>RETAILER</ns1:userType>
            <ns1:id>2</ns1:id>
        </ns0:technologist>
        <ns0:title>Caesar Salad Kit</ns0:title>
        <ns0:udfData>
            <ns0:id>84</ns0:id>
        </ns0:udfData>
        <ns0:id>1</ns0:id>
</ns0:productRecordFullDTO>
```

Error Responses

In the event that an error occurs, an HTTP 500 response is sent.

Check Record Modification Timestamp

Description

Retrieves the last modification time for a product record.

Available from Version

1.9

Endpoint Address

/services/rest/productRecord/{id}

HTTP Method

HEAD

Request Parameters

There are no request parameters, but the URL contains the {id} parameter that determines the record to load.

Response

If successful, an HTTP 200 response is sent containing the Last-Modified header to show the last modification data for that Product Record. For example:

```
HTTP/1.1 200 OK
Content-Length: 0
Expires: Thu, 01-Jan-1970 00:00:00 GMT
Set-Cookie: JSESSIONID=154osm4djwouv;Path=/creations-salesdemo
Date: Thu, 30 Apr 2015 10:33:05 GMT
Last-Modified: Fri, 28 Feb 2014 16:27:06 GMT
Server: Jetty(7.0.2.v20100331)
```

Error Responses

In the event that an error occurs, an HTTP 500 response is sent.

Create Record

Description

Creates a new Product Record.

Available from Version

1.9

Endpoint Address

/services/rest/productRecord

HTTP Method

POST

Request Body

The body of the request contains a ProductRecordFullDTO to specify the Product Record to create. For example:

```
<ns0:productRecordFullDTO
    xmlns:ns1="http://www.micros.com/creations/core/domain/dto/v1p0/simple"
    xmlns:ns0="http://www.micros.com/creations/core/domain/dto/v1p0/full">
    <ns0:businessCategory>
        <ns1:code>CATEGORY2A</ns1:code>
    </ns0:businessCategory>
    <ns0:code>1</ns0:code>
    <ns0:disciplineDetail>
        <ns0:testDiscipline>
            <ns1:code>CHEMICAL</ns1:code>
        </ns0:testDiscipline>
    </ns0:disciplineDetail>
    <ns0:disciplineDetail>
        <ns0:testDiscipline>
            <ns1:code>PHYSICAL</ns1:code>
        </ns0:testDiscipline>
    </ns0:disciplineDetail>
    <ns0:disciplineDetail>
```

```
            <ns0:testDiscipline>
                <ns1:code>MICRO</ns1:code>
            </ns0:testDiscipline>
        </ns0:disciplineDetail>
        <ns0:productCovered>
            <ns0:productTitle>Caesar Salad Kit</ns0:productTitle>
            <ns0:quantity>1</ns0:quantity>
            <ns0:retailerProductNumber>1</ns0:retailerProductNumber>
            <ns0:variantName><![CDATA[Caesar Salad Kit]]></ns0:variantName>
        </ns0:productCovered>
        <ns0:productRecordOtherUser>
            <ns0:mandatoryUser>true</ns0:mandatoryUser>
            <ns0:multipleUser>false</ns0:multipleUser>
            <ns0:role>
                <ns1:code>BUYER</ns1:code>
            </ns0:role>
            <ns0:user>
                <ns1:code>Richard Admin</ns1:code>
            </ns0:user>
        </ns0:productRecordOtherUser>
        <ns0:productRecordOtherUser>
            <ns0:mandatoryUser>true</ns0:mandatoryUser>
            <ns0:multipleUser>false</ns0:multipleUser>
            <ns0:role>
                <ns1:code>PRODUCT DEVELOPMENT MANAGER</ns1:code>
            </ns0:role>
            <ns0:user>
                <ns1:code>productdevelopmentmanager</ns1:code>
            </ns0:user>
        </ns0:productRecordOtherUser>
        <ns0:projectName>Caesar Salad Kit</ns0:projectName>
        <ns0:site>
            <ns1:code>A0001-0004</ns1:code>
        </ns0:site>
        <ns0:specTypeFormat>
            <ns1:code>FOODUK</ns1:code>
        </ns0:specTypeFormat>
        <ns0:status>
            <ns1:status>ACTIVE</ns1:status>
        </ns0:status>
        <ns0:supplier>
            <ns1:code>A0001</ns1:code>
        </ns0:supplier>
        <ns0:technologist>
            <ns1:code>techadmin</ns1:code>
        </ns0:technologist>
        <ns0:title>Caesar Salad Kit</ns0:title>
    </ns0:productRecordFullDTO>
```

Compared to the response for retrieving a Product Record (which uses the same ProductRecordFullDTO type), this request is much shorter. This is achieved by only including the information relevant to creating the Product Record. Where the record is linked to another record, such as the role, only the business keys (the code in this case) that allow that linked record to be identified are supplied. In general, this is the code attribute of the linked record, but in some cases, it is the status or a combination of values.

A Brand Compliance system allows configuration of the additional roles for which users must be nominated on a Product Record. These are held as

productRecordOtherUser records on the Product Record. The call to the web service should ensure that the entries match the system configuration.

Also configurable are the Surveillance Test Disciplines. For each Discipline active in the system, a disciplineDetail should be included with a testDiscipline populated.

Response

If successful, an HTTP 200 response is sent with a body containing a ProductRecordLink:

```
<ProductRecordLink>
   <recordId>1</recordId>
   <recordLink>
      http://localhost:9090/creations-salesdemo/services/rest/productRecord/1
   </recordLink>
   <code>1</code>
   <title>Caesar Salad Kit</title>
</ProductRecordLink>
```

The recordId is the ID of the newly created Product Record and the recordLink is a URI to the Product Record that was created for use in a GET request.

Error Responses

If the supplied data does not result in a valid Product Record (such as the status' status missing), an HTTP 417 response is sent with a body message stating the validation errors. The request should not be reattempted with the same content.

If an internal error is experienced, an HTTP 500 response is sent.

Update Record

Description

Updates an existing Product Record.

Available from Version

1.9

Endpoint Address

/services/rest/productRecord/{id}

HTTP Method

PUT

Request Body

The body of the request contains a ProductRecordFullDTO to specify how the Product Record should appear after the update:

```
<ns0:productRecordFullDTO
    xmlns:ns1="http://www.micros.com/creations/core/domain/dto/v1p0/simple"
    xmlns:ns0="http://www.micros.com/creations/core/domain/dto/v1p0/full">
    <ns0:businessCategory>
        <ns1:code>CATEGORY2A</ns1:code>
    </ns0:businessCategory>
    <ns0:code>1</ns0:code>
    <ns0:disciplineDetail>
        <ns0:testDiscipline>
```

```
                <ns1:code>CHEMICAL</ns1:code>
            </ns0:testDiscipline>
        </ns0:disciplineDetail>
        <ns0:disciplineDetail>
            <ns0:testDiscipline>
                <ns1:code>PHYSICAL</ns1:code>
            </ns0:testDiscipline>
        </ns0:disciplineDetail>
        <ns0:disciplineDetail>
            <ns0:testDiscipline>
                <ns1:code>MICRO</ns1:code>
            </ns0:testDiscipline>
        </ns0:disciplineDetail>
        <ns0:productCovered>
            <ns0:productTitle>Caesar Salad Kit-updated</ns0:productTitle>
            <ns0:quantity>1</ns0:quantity>
            <ns0:retailerProductNumber>1</ns0:retailerProductNumber>
            <ns0:variantName><![CDATA[Caesar Salad Kit-updated]]></ns0:variantName>
        </ns0:productCovered>
        <ns0:productRecordOtherUser>
            <ns0:mandatoryUser>true</ns0:mandatoryUser>
            <ns0:multipleUser>false</ns0:multipleUser>
            <ns0:role>
                <ns1:code>BUYER</ns1:code>
            </ns0:role>
            <ns0:user>
                <ns1:code>Richard Admin</ns1:code>
            </ns0:user>
        </ns0:productRecordOtherUser>
        <ns0:productRecordOtherUser>
            <ns0:mandatoryUser>true</ns0:mandatoryUser>
            <ns0:multipleUser>false</ns0:multipleUser>
            <ns0:role>
                    <ns1:code>PRODUCT DEVELOPMENT MANAGER</ns1:code>
            </ns0:role>
            <ns0:user>
                <ns1:code>productdevelopmentmanager</ns1:code>
            </ns0:user>
        </ns0:productRecordOtherUser>
        <ns0:projectName>Caesar Salad Kit-updated</ns0:projectName>
        <ns0:site>
            <ns1:code>A0001-0004</ns1:code>
        </ns0:site>
        <ns0:specTypeFormat>
            <ns1:code>FOODUK</ns1:code>
        </ns0:specTypeFormat>
        <ns0:status>
        <ns1:status>ACTIVE</ns1:status>
        </ns0:status>
        <ns0:supplier>
            <ns1:code>A0001</ns1:code>
        </ns0:supplier>
        <ns0:technologist>
        <ns1:code>techadmin</ns1:code>
        </ns0:technologist>
        <ns0:title>Caesar Salad Kit-updated</ns0:title>
        <ns0:id>1</ns0:id>
    </ns0:productRecordFullDTO>
```

The request content is the same as that for creating a Product Record. After the call, the Product Record is updated to match the request. If a value is omitted from the request, the value is un-set on the Product Record.

Response

If successful, an HTTP 200 response is sent with a body containing a ProductRecordLink:

```
<ProductRecordLink>
   <recordId>1</recordId>
   <recordLink>
      http://localhost:9090/creations-salesdemo/services/rest/productRecord/1
   </recordLink>
   <code>1</code>
   <title>Caesar Salad Kit-updated</title>
</ProductRecordLink>
```

The recordId is the ID of the updated Product Record and the recordLink is a URI to the Product Record record that was created for use in a GET request.

Error Responses

If the supplied data does not result in a valid Product Record (such as the status's status missing), an HTTP 417 response is sent with a body message stating the validation errors. The request should not be reattempted with the same content.

If an internal error is experienced, an HTTP 500 response is sent.

ProductSpecificationRestService

This section describes the API for managing product specifications.

List of Values

Description

Retrieves a list of product specifications in a paged list.

Available from Version

1.9

Endpoint Address

/services/rest/specification

HTTP Method

GET

Request Parameters

Parameters are passed as URI parameters.

URI Parameters

Parameter Name	Mandatory/Optional	Value Type	Example
offset	Optional	Numeric	60

Parameter Name	Mandatory/Optional	Value Type	Example
pageSize	Optional	Numeric	30

The offset and pageSize parameters control the paging of the list. The offset defines the first record to retrieve (0-based) and the pageSize defines how many records to include in the list.

Response

For a successful response, XML is returned with a ProductSpecificationLinkList root element containing an entries element for each matched Product Specification with the specification number, specification version, title and a recordLink URI to the ProductSpecificationRestService Retrieve record service for this product specification. For example:

```
<ProductSpecificationLinkList>
    <entries xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:type="productSpecificationLink">
            <recordId>2</recordId>
        <recordLink>
           http://localhost:9090/creations-salesdemo/services/rest/specification/2
        </recordLink>
        <specNumber>5</specNumber>
        <specVersion>1</specVersion>
            <title>Caesar Salad</title>
    </entries>
    <nextPage>
http://localhost:9090/creations-salesdemo/services/rest/specification/?offset=1&am
p;pageSize=1
    </nextPage>
        <totalRecords>173</totalRecords>
    <entryArray>
        <recordId>2</recordId>
        <recordLink>
           http://localhost:9090/creations-salesdemo/services/rest/specification/2
        </recordLink>
            <specNumber>5</specNumber>
        <specVersion>1</specVersion>
        <title>Caesar Salad</title>
    </entryArray>
</ProductSpecificationLinkList>
```

Error Responses

In the event that an error occurs, an HTTP 500 response is sent.

Retrieve Record

Description

Retrieves a single product specification's details.

Available from Version

1.9

Endpoint Address

/services/rest/specification/{id}

HTTP Method
GET

Request Parameters
There are no request parameters, but the URL contains the {id} parameter that determines the record to load.

Response
For a successful response, XML is returned with a productSpecificationFullDTO root element containing the details of the Product Specification.

No example is included for Product Specification due to the large size of such an example.

Error Responses
In the event that an error occurs, an HTTP 500 response is sent.

Check Record Modification Timestamp

Description
Retrieves the last modification time for a product specification.

Available from Version
1.9

Endpoint Address
/services/rest/specification/{id}

HTTP Method
HEAD

Request Parameters
There are no request parameters, but the URL contains the {id} parameter that determines the record to load.

Response
If successful, an HTTP 200 response is sent containing the Last-Modified header to show the last modification data for that Product Specification. For example:

```
HTTP/1.1 200 OK
Content-Length: 0
Expires: Thu, 01-Jan-1970 00:00:00 GMT
Set-Cookie: JSESSIONID=154osm4djwouv;Path=/creations-salesdemo
Date: Thu, 30 Apr 2015 10:33:05 GMT
Last-Modified: Fri, 28 Feb 2014 16:27:06 GMT
Server: Jetty(7.0.2.v20100331)
```

Error Responses
In the event that an error occurs, an HTTP 500 response is sent.

Retrieve List with Advanced Filtering

Description

Retrieves a list of product specifications in a paged list using advanced filtering. This is a RESTful equivalent of the getProductSpecificationServiceV1 SOAP service.

Available from Version

1.9

Endpoint Address

/services/rest/specification/advanced

HTTP Method

POST

Request Body

The filters for the request are passed in using an XML Body:

```
<GetProductSpecificationServiceRequest>
    <batchsize>?</batchsize>
    <offset>?</offset>
    <!--Zero or more repetitions:-->
    <specStatus>?</specStatus>
    <!--Zero or more repetitions:-->
    <specType>?</specType>
    <!--Zero or more repetitions:-->
    <countryWhereSold>?</countryWhereSold>
    <!--Zero or more repetitions:-->
    <productNumber>?</productNumber>
    <!--Zero or more repetitions:-->
    <language>?</language>
    <!--Zero or more repetitions:-->
    <sectionType>?</sectionType>
    <!--Zero or more repetitions:-->
    <productCoverage>?</productCoverage>
    <!--Zero or more repetitions:-->
    <supplier>?</supplier>
    <!--Zero or more repetitions:-->
    <site>?</site>
    <!--Optional:-->
    <fromDate>?</fromDate>
    <!--Optional:-->
    <toDate>?</toDate>
</GetProductSpecificationServiceRequest>
```

For more details on the use and impact of these request filters, see the documentation for the getProductSpecificationServiceV1 SOAP service.

Response

For a successful response, XML is returned with a GetProductSpecificationServiceResponse root element.

Due to the length of the response, an example is not included.

Error Responses

In the event that an error occurs, an HTTP 500 response is sent.

TaskRestService

This section describes the API for managing user tasks.

List of Tasks

Description

Retrieves a list of tasks for the supplied user in the given language.

Available from Version

1.9

Endpoint Address

/services/rest/task

HTTP Method

GET

Request Parameters

Parameters are passed as URI parameters.

URI Parameters

Parameter Name	Mandatory/Optional	Value Type	Example
userId	Mandatory	String	testuser
language	Mandatory	String	en_GB

The userId parameter is the login ID of the user for which the tasks list is to be retrieved. The language parameter is the code of the language record/locale in which to retrieve the task details.

Response

For a successful response, XML is returned with a TaskDTOList root element containing a tasks element for each matched Task with the message (in the language specified), messageId (a language-agnostic identifier for the task), taskItemCount (the number of items in that task), and the myCreations Link which is a URI back to the Brand Compliance Management Cloud Service system which will open the list of items for that task. For example:

```
<TaskDTOList>
    <tasks>
        <message>You have 1 document(s) requiring attention</message>
        <messageId>tasks.documentsToRead</messageId>
        <myCreationsLink>
http://localhost:9090/creations-salesdemo/app?flow=com.micros.creations.core.domai
n.DocumentHeaderRetailerList&amp;amp;task=95&amp;amp;user=producttechnologist
        </myCreationsLink>
        <taskItemCount>1</taskItemCount>
    </tasks>
</TaskDTOList>
```

Error Responses

If both userId and language are missing from the request, an HTTP 417 response is sent with an ErrorMessage XML body:

```
<ErrorMessage><message>userId and language required</message></ErrorMessage>
```

If the userId parameter is missing, an HTTP 417 status is returned with an ErrorMessage XML body:

```
<ErrorMessage><message>userId required</message></ErrorMessage>
```

If the language parameter is missing, an HTTP 417 status is returned with an ErrorMessage XML body:

```
<ErrorMessage><message>language required</message></ErrorMessage>
```

If there is an internal error, an HTTP 500 response is sent.

UrgentItemsRestService

This section describes the API for managing urgent items for users.

Number of Urgent Items

Description

Retrieves the number of urgent items pending for a given user.

Available from Version

1.9

Endpoint Address

/services/rest/urgentItems

HTTP Method

GET

Request Parameters

Parameters are passed as URI parameters.

URI Parameters

Parameter Name	Mandatory/Optional	Value Type	Example
userId	Mandatory	String	testuser

The userId parameter is the login ID of the user for which the number of pending urgent items are to be retrieved.

Response

For a successful response, XML is returned with an UrgentItemsModel root element containing an itemCount element that specifies the number of Urgent Items pending for that given user. For example:

```
<UrgentItemsModel><itemCount>0</itemCount></UrgentItemsModel>
```

Error Responses

If the userId parameter is missing, an HTTP 417 status is returned with an ErrorMessage XML body:

```
<ErrorMessage><message>userId required</message></ErrorMessage>
```

If there is an internal error, an HTTP 500 response is sent.

DataPrivacyService

This section describes the API for executing Data Privacy requests. The following functions are available:

- Right to Access: retrieves personal data from the system relating to the name of an individual
- Right to be Forgotten: erases personal data from the system for an individual

Although minimal personal data is held in Brand Compliance, it does include the names and contact details of retailer/portal owner, supplier, and third-party users. The Data Privacy API provides a means for the retailer/portal owner to:

1. Request details of personal data relating to an individual be held within the system.
2. Request that the personal data relating to the individual be removed from the system.

For additional detail on using the Data Privacy API, including examples, see Appendix B, "Appendix: Using the Data Privacy API."

Right to Access

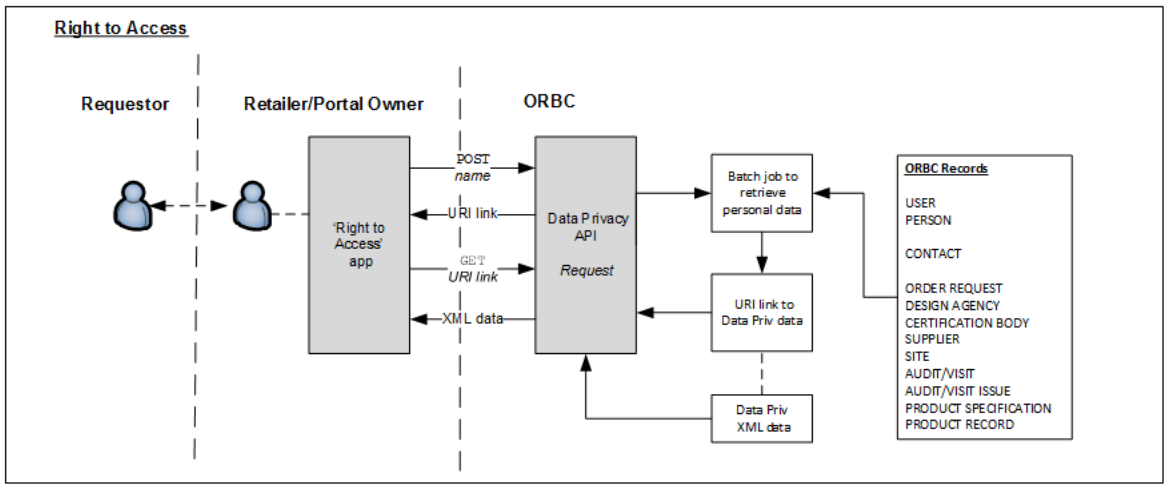
Description

Retrieves a set of personal data that relates to the name of an individual. Use this function to perform Right to Access requests for the following scenarios:

- An employee of the retailer/portal owner requests an electronic copy of their personal data that is held within the system.
- An employee of one of the retailer/portal owner's suppliers requests an electronic copy of their personal data that is held within the system.
- An employee of the retailer/portal owner's partner organizations requests an electronic copy of their personal data that is held within the system.

The API's access function allows the name of the individual to be passed, with an XML message being returned containing any personal data found for that name. The XML message is a structured text format, which is both machine and human readable.

Figure 4–1 Right to Access Process



The steps for completing a Right to Access request are as follows (for further details, see Appendix B, "Appendix: Using the Data Privacy API"):

1. Submit a POST call, passing the name of the individual.

   A batch job is submitted within Brand Compliance to search the system for personal data for the presence of the name (for which records and fields are searched, see Appendix B, "Appendix: Using the Data Privacy API"). On completion, a Data Privacy record containing the retrieved personal data is created within Brand Compliance.

   A link to the Data Privacy record is returned as the response to the POST call.

2. On receipt of the response to the POST call, submit a GET call, passing the returned URI link.

   Note: If the GET call is submitted before the batch job has finished compiling the personal data, just a header will be returned, and the GET call should be retried.

   A message containing the personal data is returned. The XML data is structured within a dataprivRequestDTO root element (for details of the message structure, see Appendix B, "Appendix: Using the Data Privacy API").

   Just the personal data fields and values to identify the record are returned, not the full record contents.

   Addresses may contain multiple localeData elements for the language translation of countries.

   XML elements will only be present if they contain data.

3. Assess whether the returned data relates to the individual who made the request, editing and/or reformatting it if necessary, before passing it to the requestor.

Endpoint address: /services/rest/datapriv/access
HTTP methods: POST, GET

Request Details

For the POST method, a name parameter is passed in the payload.

For the GET method, an id parameter is passed as a URI parameter.

URI Parameters

Example URL (POST method):
…/services/rest/access/

Example URL (GET method):
…/services/rest/access/?id=16

Response Details

For a successful response to the POST method, a link to the Data Privacy record is returned.

For a successful response to the GET method, XML is returned with a dataprivRequestDTO root element containing an element for each record where the personal data has been found.

Error Responses

In the event that an error occurs, an HTTP 500 response is sent.

Right to be Forgotten

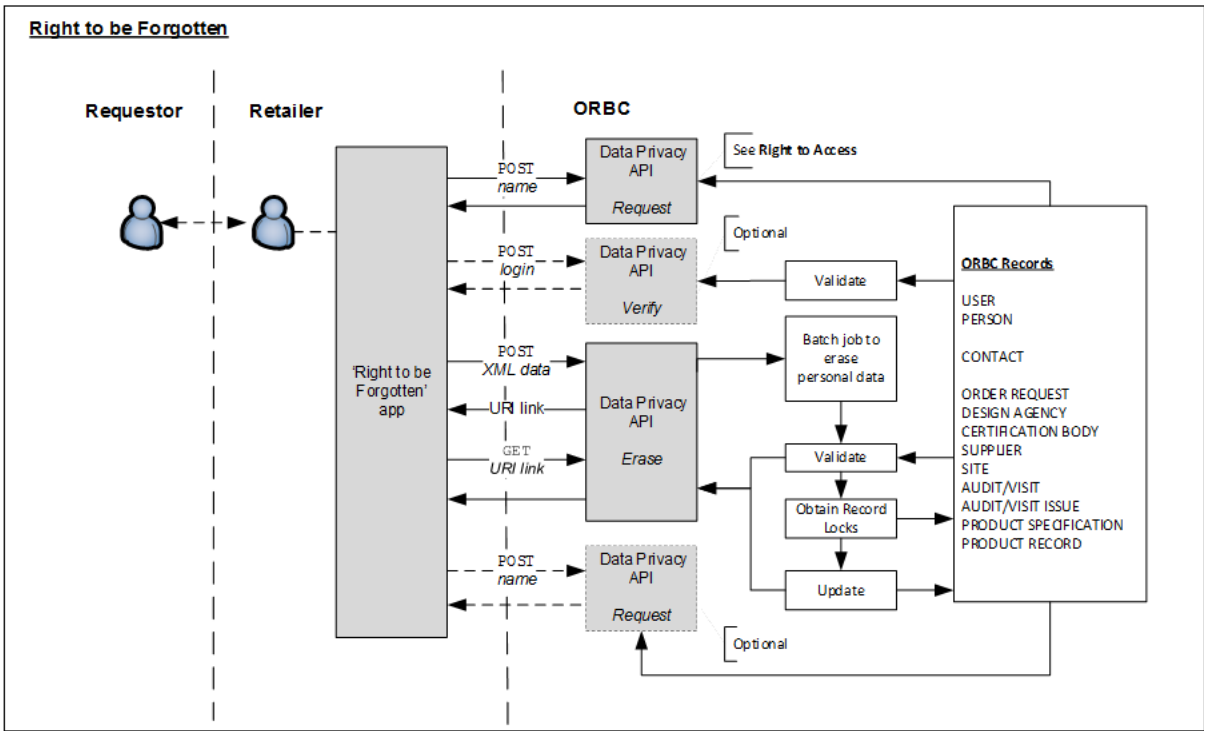
Description

Erases a set of personal data that relates to an individual. Use this function to perform Right to be Forgotten requests for the following scenarios:

- An employee of the retailer/portal owner requests their personal data be erased from the system.
- An employee of one of the retailer/portal owner's suppliers requests their personal data be erased from the system.
- An employee of a partner organizations requests their personal data be erased from the system.
- The retailer/portal owner initiates the purging of an inactive user's personal data from the system.

An erase function allows the XML message returned from a Right to Access request to be used to erase the data.

Figure 4–2 Right to be Forgotten Process



The steps for completing a Right to be Forgotten request are as follows:

1. The Right to Access service must be used to retrieve the personal data for the individual. For details, see "Right to Access".
2. View, and if necessary, edit the XML returned from the Right to Access request.

   As there may be any number of users with the same name, it is necessary to decide which data represents the individual who made the request, removing any that is not relevant (for further information on editing the XML message, see Appendix B, "Appendix: Using the Data Privacy API").
3. Optionally submit a POST call to the verify function, passing the login id of the individual (obtained from the XML message returned in Step 2).

   This step is used to check whether the erasure of data for the individual will be permitted, before actually submitting the request. The erase will not be permitted if the individual is responsible for the completion of a supplier site's Scorecard.

   XML is returned with a dataprivVerifyDTO root element containing a status and a message element. If the call is successful the message will contain *Success*, otherwise it will contain *Failure* and the reason for the failure will be present in the message element.
4. Submit a POST call to the erase function, passing the XML retrieved from the Right to Access request (with any edits having been applied).

   A batch job is submitted to validate the request, verify that the erasure is permitted (that is, the individual is not responsible for the completion of a Scorecard), and obtain locks on the records to be updated.
5. If the validation fails, or a lock cannot be obtained, the process is terminated and an XML is returned with a dataprivErasureDTO root element containing a

message element that contains the reason for the failure, and the status element set to *Failure*.

If no exceptions are encountered, a Data Privacy record containing the personal data to erase is created and a link to the record is returned as the response to the POST call.

6. On receipt of the response to the POST call, submit a GET call to the erase function, using the returned URI link.

   The update processing replaces the data passed in the XML message with 30 characters of randomly-generated text to anonymize it. Email addresses are changed to x@example.com, where x is 15 characters of randomly-generated text*. GPS coordinate numbers and country selections are blanked.

   Some fields have a specific length, in which case the 30 characters of obfuscated text is overridden:

   - Name 50 characters
   - Login 20 characters
   - Address lines 20 characters
   - Post Code 5 characters

   * Email addresses that are stored as free text are fully obfuscated, rather than set to x@example.com.

   Only certain fields are permitted to be erased (for details, see Appendix B, "Appendix: Using the Data Privacy API").

   When updating a user account, the account is set to deactivated, and the password is set to expired.

   Note: If the GET call is submitted before the batch job has finished compiling the personal data, just a header will be returned, and the GET call should be retried.

   The erasure of data is not reversible.

7. Optionally submit another Right to Access request to check that the data has been erased as expected.
8. Inform the requestor of the outcome of their request.

The Right to be Forgotten facility may also be used for purging user accounts that are no longer active.

Endpoint address: /services/rest/datapriv/erase
HTTP methods: POST (verify and erase), GET (erase)

Request Details

For the verify POST method, a loginId parameter is passed in the payload.

For the erase POST method, an XML message is passed in the payload.

For the erase GET method, an id parameter is passed as a URI parameter.

URI Parameters

Example URL (verify POST method):
…/services/rest/verify/

Example URL (erase POST method):
…/services/rest/erase/

Example URL (erase GET method):
…/services/rest/erase/?id=32

Response Details

For a successful response to the verify POST method, XML is returned with a dataprivVerifyDTO root element containing a status of *Success*.

For a successful response to the erase POST method, a link to the Data Privacy record is returned.

For a successful response to the erase GET method, an HTTP 200 response is sent with a dataprivErasureDTO root element containing a status of *Success*.

Error Responses

For a failure response to the verify method, XML is returned with a dataprivVerifyDTO root element containing a status of *Failure*, with the reason in the message element.

For a failure response to the erase method, XML is returned with a dataprivErasureDTO root element containing a status of *Failure*, with the reason in the message element.

Error Messages

Element	Message	Meaning
user	Only one User can be erased at a time	More than one User was passed in to the erase function. The XML must only relate to a single user.
person	Only one Person can be erased at a time	More than one Person was passed in to the erase function. The XML must only relate to a single person.
id	The User is linked to Person with ID: %id but the Person included has ID: %id	The wrong Person has been included. %id is the id taken from the XML.
id	The Contacts are linked to Person with ID: %id but the Person included has ID: %id	The wrong Person has been included. %id is the id taken from the XML.
id	The Contact with ID %id is linked to a different User than the one being erased	The Contacts passed do not match the user being erased. %id is replaced with the id from the XML.
person	All Contacts being erased must be linked to the same Person	Where no User is being erased, all Contacts being erased must be for the same person.
person	Please include Person with ID: %id	If User or Contacts are passed, the matching Person must also be included.

Element	Message	Meaning
id	Cannot find entity of type: %rec and ID: %id	An id is passed that is not found. %rec is the internal name of the record, such as ProductRecord. %id is the id taken from the XML.
id	Entity of type: %rec and ID: %id is currently locked, please try again later	One of the records to be erased is currently locked. %rec is the internal name of the record, such as ProductRecord. %id is the id taken from the XML. Retry the request until the record becomes available.
id	Entity of type: %rec with id: %id has been edited during the erase process, please verify data	One of the records updated has been edited between up front validation and the processing of that particular record. %rec is the internal name of the record, such as ProductRecord. %id is the id taken from the XML.
user	The user cannot be erased because they are responsible for the approval of a Scorecard	The user is named as being responsible for completing a scorecard, so they cannot be erased.
login id	Cannot find User with login ID: %id	No User account found for the passed login id. %id is the login id.
	The erase has failed due to unexpected internal error	Other unexpected condition has occurred.

A

Appendix: Testing and Developing Clients for APIs

This appendix provides additional information about testing and developing the clients for APIs.

Using SoapUI for Testing the APIs

SoapUI is a very useful tool for testing APIs exposed as web services. There is an Open Source version that allows the invocation of both RESTful and SOAP architecture services. For information on this tool, see the following web site:

http://www.soapui.org/

The following sections describe how to use SoapUI with the Brand Compliance APIs. Installation instructions are not included as these will vary greatly depending on the version of SoapUI and the installation platform in use.

Testing a SOAP Service

To test a SOAP service:

1. Download the WSDL for the service endpoint that you wish to call. You can see the list of services at /services. Each service includes a link to the WSDL file that you can right-click on to download.
2. With SoapUI open, take the File ? New SOAP Project option to open the New SOAP Project dialogue. Enter a suitable Project Name and use the Browse... button to select the WSDL file that you saved earlier. Click OK.
3. You may be prompted to enter a Username and Password multiple times if the WSDL includes references to separate schema files. In these prompts, you need to enter the login ID and password for an External System in Brand Compliance.
4. After some time, a new project appears in the Project navigator view of the SoapUI window with a Binding node, Operation node, and a Request node. If you double-click the Request node, a window opens showing a sample request ready for you to fill in with the data.
5. Within the Request window, there is an Auth button that when clicked shows an Authorization view. Change the drop-down in this view to Add New Authorisation... to open the Add Authorization dialog. In this dialog, select Type as Basic and click OK. Enter the External System information into the Username and Password fields that have appeared in the Authorization view, and change Pre-emptive auth to Authenticate pre-emptively.

6. Once you have entered appropriate values in the request XML, click the green arrow button on the top-left of the Request window to make the request. The response is visible on the right-hand side.

Testing a RESTful Service

To test a RESTful service:

1. Download the WADL for the service containing the Endpoint that you wish to call. You can see the list of services at /services. Each service includes a link to the WADL file that you can right-click on to download.
2. With SoapUI open, take the File ? New REST Project action to open up the New REST Project dialog. Within the dialog, click the Import WADL... file to change the dialog to the New WADL Project dialog. Select the downloaded WADL using the Browse... button and click OK.

   After some time, a new project appears in the Project navigator view of the SoapUI window with a Binding node and nested Operation nodes, each with suitable HTTP Method nodes and each with Request nodes under them.
3. Double click on the Request node for the service that you wish to invoke. If there are parameters to set, they appear as a list in the new Request window. The Value column should be populated with any matching values that you wish to set.
4. Within the Request window, there is an Auth button, that when clicked, shows an Authorization view. Change the drop-down in this view to Add New Authorisation... to open the Add Authorization dialog. In this dialog, select Type as Basic and click OK. Enter the External System information into the Username and Password fields that appear in the Authorization view, and change Pre-emptive auth to Authenticate pre-emptively.
5. If the Request includes XML as part of the request, an additional area is visible for you to enter the XML. Also, a drop-down field for Media Type should be visible that needs to be set to application/XML.
6. Once you have entered appropriate values in the request XML and/or parameters, click the green arrow button on the top-left of the Request window to make the request. The response is visible on the right-hand side.

Creating Clients Using Apache CXF

Apache CXF is a popular Web Services framework for Java that allows clients to be generated for both SOAP and RESTful architecture APIs.

Creating a Client for a SOAP Service

It is assumed for the following steps that you have a project using maven as a project management tool. If you do not, you need to go the following web site for a more general approach that is not using Maven to execute the wsdl2java tool:

http://cxf.apache.org/docs/developing-a-consumer.html

To create a client for a SOAP service:

1. Create a wsdl directory in your project's src/main/resources directory.
2. Download the WSDL from the service that you want to invoke, for example:

   https://www.example.com/brandCompliance/services/productService?wsdl

3. Place it into the src/main/resources/wsdl directory of your project. Inspect the WSDL to check for any references to schemas or WSDL definition files if you see a line such as the following:

```
<wsdl:import
location="http://localhost:9090/creations-salesdemo/services/productService?wsd
l=ProductService.wsdl" namespace="external.service.creations.micros.com">
```

or

```
<import namespace="v1.getproductspecification.service.creations.micros.com"
schemaLocation="http://localhost:9090/creations-salesdemo/services/getProductSp
ecificationV1?xsd=GetProductSpecificationServiceV1_schema1.xsd"/>
```

4. Download the file from the URI given in the location or schemaLocation attribute and place it into the src/main/resources/wsdl folder of your project. Change the location/schemaLocation attribute to the name of the downloaded file. Repeat this for the downloaded files in case they also point to other server-based resources.

5. In your pom.xml, add a plug-in definition to execute the wsdl2java tool for that WSDL:

```
<plugin>
    <groupId>org.apache.cxf</groupId>
    <artifactId>cxf-codegen-plugin</artifactId>
    <version>${cxf.version}</version>
    <executions>
        <execution>
            <id>generate-sources</id>
            <phase>generate-sources</phase>
            <configuration>
                <sourceRoot>
                    ${project.build.directory}/generated/cxf
                </sourceRoot>
                <wsdlOptions>
                    <wsdlOption>
                        <wsdl>
                         ${basedir}/src/main/resources/wsdl/productService.wsdl
                        </wsdl>
                        <extraargs>
                            <extraarg>-client</extraarg>
                        </extraargs>
                    </wsdlOption>
                </wsdlOptions>
            </configuration>
            <goals>
                <goal>wsdl2java</goal>
            </goals>
        </execution>
    </executions>
</plugin>
```

Note: You need to define a build property cxf.version to specify the version of Apache CXF to use:

```
<properties>
   <cxf.version>2.7.5</cxf.version>
</properties>
```

Running mvn clean generate-sources generates the client stubs into your project under /target/generated-sources/cxf including a class name that ends with _Client that contains a main method giving an example of calling the service.

By default, the invocation of the web service uses the WSDL in the project based on its file location. To work when deployed into a container, you need to reference the WSDL as a classpath resource. Additionally, the endpoint address of the service to invoke may need to be configurable in your client so that you can use the same code for your testing as you would in live. Following is an example of calling a service using a classpath-retrieved WSDL file and a specific endpoint address:

```
URI wsdlURL =
Thread.currentThread().getContextClassLoader().findResources("wsdl/productService.
wsdl");
ProductServiceImplService ss = new ProductServiceImplService(wsdlURL);
ProductService port = ss.getProductServiceImplPort();

BindingProvider bindingProvider = (BindingProvider) port;
bindingProvider.getRequestContext().put(
      BindingProvider.ENDPOINT_ADDRESS_PROPERTY,
      "https://www.example.com/brandCompliance/productService");

CreateProductRequest req = null;

// create and populate the CreateProductRequest

CreateProductResponse = port.createProduct(req);
```

Creating a Client for a RESTful Service

It is assumed for the following steps that you have a project using maven as a project management tool. If you do not, you need go to the following web site for documentation on how to execute the wadl2java tool outside of maven:

http://cxf.apache.org/docs/jaxrs-services-description.html#JAXRSServicesDescription-WADL-firstDevelopment

To create a client for a RESTful service:

1. Create a config directory in your project's src/main directory.
2. Download the WADL file for the service that you wish to invoke. For example: https://www.example.com/brandCompliance/services/rest/user?_wadl
3. Place it into the src/main/config directory of your project.
4. In your project's pom.xml file, add a property for the version of Apache CXF you wish to use:

```
<properties>
  <cxf.version>2.7.5</cxf.version>
</properties>
```

5. Add the required compile and runtime dependencies:

```
<dependencies>
  <dependency>
      <groupId>org.apache.cxf</groupId>
      <artifactId>cxf-rt-frontend-jaxrs</artifactId>
      <version>${cxf.version}</version>
  </dependency>
      <dependency>
```

```
            <groupId>org.springframework</groupId>
            <artifactId>spring-core</artifactId>
            <version>3.2.1.RELEASE</version>
            <scope>compile</scope>
        </dependency>
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-web</artifactId>
            <version>3.2.1.RELEASE</version>
        </dependency>
    </dependencies>
```

6. Add a plug-in to execute the wadl2java tool:

```
<build>
<plugins>
<plugin>
  <groupId>org.apache.cxf</groupId>
  <artifactId>cxf-wadl2java-plugin</artifactId>
  <version>${cxf.version}</version>
  <executions>
      <execution>
          <id>generate-sources</id>
          <phase>generate-sources</phase>
          <configuration>
              <sourceRoot>
                  ${basedir}/target/generated/src/main/java
              </sourceRoot>
              <wadlOptions>
                  <wadlOption>
                      <wadl>${basedir}/src/main/config/user.wadl</wadl>
                      <packagename>
                          com.micros.creations.rest.service
                      </packagename>
                      <extraargs>
                          <extraarg>-resource</extraarg>
                          <extraarg>UserRestService</extraarg>
                      </extraargs>
                  </wadlOption>
              </wadlOptions>
          </configuration>
          <goals>
              <goal>wadl2java</goal>
          </goals>
      </execution>
  </executions>
</plugin>
</plugins>
</build>
```

7. Running mvn clean generate-sources should now create JAXB annotated types in target/generated/src/main/java within the project. Due to an issue with the way that wadl2java creates the XmlElementDecl annotations in the ObjectFactory, you need to edit the file target/generated/src/main/java/com/micros/creations/core/domain/dto/v1p0/simple/ObjectFactory.java. For all methods whose name matches create*LinkList, edit the XmlElementDecl to set the namespace to the empty string - "".

```
/**
```

```
 * Create an instance of {@link JAXBElement }{@code <}{@link UserList }{@code
>}}
 *
 */
@XmlElementDecl(namespace = "", name = "UserLinkList")
public JAXBElement<UserList> createUserLinkList(UserList value) {
return new JAXBElement<UserList>(_UserLinkList_QNAME, UserList.class, null,
value);
    }

With the set-up of the project complete, we can now invoke a service using the
JAXRSClientFactory:
String username;
String password;

// load username and password from a secure store

UserRestService userService = JAXRSClientFactory.create(
"https://www.example.com/brandCompliance/services/rest/user",
UserRestService.class, username, password,
(String) null);

// retrieve up to 30 users starting at the first user
Response r = userService.get(0, 30);
r.bufferEntity();
Object o = r.getEntity();
UserList list = null;

if (o instanceof InputStream) {
try {
JAXBContext jc = JAXBContext.newInstance(
  "com.micros.creations.core.domain.dto.v1p0.full:"
+ "com.micros.creations.core.domain.dto.v1p0.simple");
JAXBElement listElement = (JAXBElement) jc.createUnmarshaller()
.unmarshal((InputStream) o);
Object val = listElement.getValue();
if (val instanceof UserList) {
list = (UserList) val;
} else {
throw new IllegalStateException("Received a value that is not a UserList");
}
for (UserLink link : list.getEntryArray()) {
System.out.println("Found user " + link.getLoginId());
}
} finally {
((InputStream) o).close();
}
} else {
throw new IllegalStateException("Returned Entity is not an InputStream");
}
```

B

Appendix: Using the Data Privacy API

The Brand Compliance Data Privacy web service API provides a means for the retailer/portal owner to execute data privacy requests. The requests are classified as Right to Access, where an individual requests details of all their personal data that is held in the system; and Right to be Forgotten, where the individual requests that their personal data be erased from the system.

> **Note:** The provision of the Data Privacy API is a security enhancement which aids the retailer/portal owner to carry out data privacy requests.
>
> It is the responsibility of the retailer/portal owner to manage the fulfilment of data privacy requests. In order to do so, they will need to build an application to call the Data Privacy API, and handle the returned data. The API will be permanently available for access on demand.
>
> The retailer/portal owner's responsibilities include verifying the identity of the individual making the request, presenting the information back to them in a secure manner, and deleting any local copies of the data; also the configuration of the portal's Terms and Conditions to include statements about the consent to store personal data, and any opt-out procedure.

This appendix provides additional detail to support the DataPrivacyService section of this document. It covers the following areas:

- Definition of Personal Data: an explanation of some of the terminology used, and the type of data within Brand Compliance that is considered to be personal data.
- Personal Data Flow: an overview of the flow of data within Brand Compliance which is considered to be personal data.
- Right to Access Requests: further detail on this type of request, with an example of its execution.
- Right to be Forgotten Requests: further detail on this type of request, with an example of its execution.
- Other Aspects of Data Privacy: other aspects of Data Privacy and how they relate to Brand Compliance.

Definition of Personal Data

The following terms are used to describe the actors involved in the handling and processing of personal data:

- Data Subject - The person whom the personal data describes. In Brand Compliance, this refers to an employee of the retailer/portal owner, an employee of a supplier, or an employee of a third party such as a design agency or certification body.
- Data Controller - The retailer/portal owner. In a cloud environment, Oracle becomes the data custodian, but the retailer/portal owner retains ownership and control of the data.
- Data Processor - Applications that work with personal data, such as Brand Compliance, and any external applications that share its data.

The definition of personal data is any information that can be mapped to a unique user, whether the identifiable information is stored with the key or not.

The personal data held in Brand Compliance is business contact information, such as:

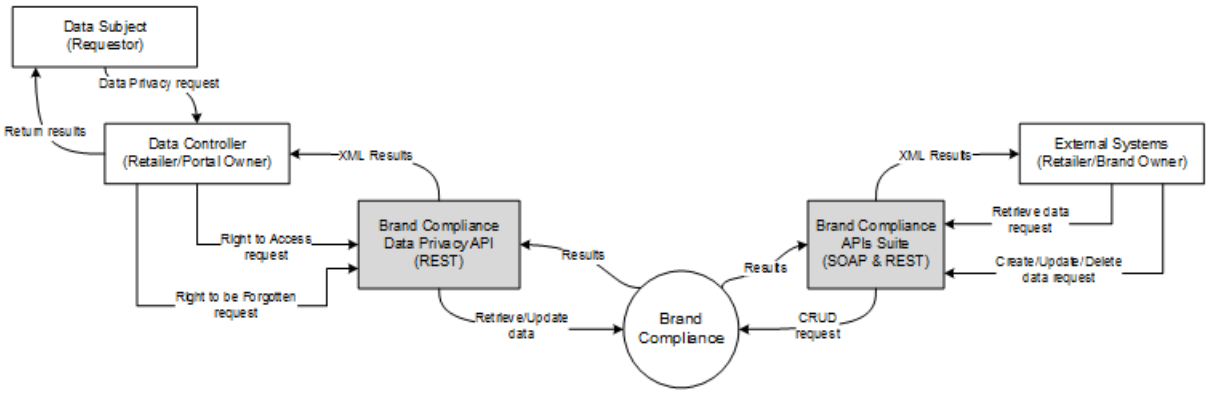
- Name of the data subject
- Contact addresses
- Email addresses
- Telephone and Fax numbers

A general assumption is that user-defined/custom fields and file attachments are not used to store personal data. This is applied as a policy constraint rather than specific controls within the system.

Personal Data Flow

The following diagram outlines the flow of personal data within Brand Compliance:

Figure B–1 Personal Data Flow



The Data Subject (requestor) makes the request to the Data Controller (retailer/portal owner) who submits the request to the Brand Compliance Data Privacy API. The API performs the search or erase processing accordingly and returns the results for the retailer/portal owner to relay back to the requestor.

The full suite of Brand Compliance APIs allows the retailer/portal owner to make requests from external systems to retrieve, create, update, (and in some cases delete) Brand Compliance data. The data handled by these APIs may include personal data.

Access to Brand Compliance data through the APIs is secured by an extension of the system's configurable Permissions security model. Based on the role-based access control model which restricts users' access to data and functionality based on their role within the system, external systems are granted access to the API services. If necessary, the access can be restricted to individual endpoints of the services. Authentication is achieved using individual ids and passwords.

Note: Brand Compliance does not *push* data to external systems; the retailer/portal owner *pulls* the data. It is therefore the responsibility of the retailer/portal owner to trace and flag any personal data that has been taken from Brand Compliance, as appropriate.

Right to Access Requests

The Right to Access service facilitates the request from the data subject (individual requestor) to the data controller (retailer/portal owner) to provide details of what personal data concerning them is held, and to provide a copy of the data in electronic format.

The majority of personal data is held within User/Person records (an account for each user of the system) and Contact records (supplier users may be designated specific contact roles). There are also instances of personal data that are not linked to a user account, such as the names and email addresses of general contacts, who are not users of the system. The retailer/portal owner must decide which of that data returned by a search relates to the requestor.

When a retailer/portal owner receives a request, they use the *Right to Access* application they have developed to call the Data Privacy API. The API searches Brand Compliance and returns the results for the retailer/brand owner to analyze, and edit or reformat if necessary, before relaying back to the requestor.

Personal Data Searched and Returned

The following table describes the areas of the system that hold personal data, and which fields are searched and returned.

Personal Data

Record/ Element Name	Details
User <user>	Each user of the system has a User record and a Person record which contain their Name, Email, Phone, Mobile, Fax, and Address details. Each account has a Login id, which is unique to the user. Names and email addresses are not necessarily unique. **Search Field:** Name **Returned Fields:** Record Type; Name; Email; Phone; Mobile; Fax; Login; Address; User Type (RETAILER, SUPPLIER or SITE); User record id; Person record id

Record/ Element Name	Details
Contact <contact>	A supplier user may be designated a contact for the supplier or its sites. Contact records share data with the User and Person records; they contain the Name, Email, Alternative Email, Phone, Mobile, Fax, and Address fields. A contact may be an individual who is not a user of the system. **Search Field:** Name (where contact is flagged as not a user of the system) **Returned Fields:** Record Type; Name; Email; Alternative Email; Phone; Mobile; Fax; Address; Contact record id; Supplier Name; Supplier Code
Company <retailer>	The single Company record holds the main retailer/portal owner contact details of a Contact Name and associated Email, Phone, Fax, and Address fields. The contact may or may not be a user of the system. **Search Field:** Contact Name **Returned Fields:** Record Type; Contact Name; Email; Phone; Fax; Address; Company record id
Order Request <orderRequest>	Order Requests are created when a new supplier completes registration. They contain details entered using the registration wizard, such as the name of an Individual or Department, a Contact Name, and associated Email, Phone, Fax, and Address. The contact may or may not be a user of the system. **Search Field:** Individual or Department; Contact Name **Returned Fields:** Record Type; Individual or Department; Contact Name; Email; Phone; Fax; Address; Order Request record id; Supplier Name; Supplier Code
Design Agency <designAgency>	A glossary of Design Agencies is used for sending email containing Pack Copy files when a product specification is completed. Each record contains a Contact Name and supporting Email, Phone, Fax, and Address, which refers to an individual at a third-party organization. **Search Field:** Contact Name **Returned Fields:** Record Type; Contact Name; Email; Phone; Fax; Address; Design Agency record id
Counter Ticket Producer <counterTicketProducer>	A glossary of Counter Ticket Producers is used for sending email containing Counter Ticket files when a Food or Produce counter ticket specification is completed. Each record contains a Contact Name and supporting Email, Phone, Fax, and Address, which refers to an individual at a third-party organization. **Search Field:** Contact Name **Returned Fields:** Record Type; Contact Name; Email; Phone; Fax; Address; Counter Ticket Producer record id
Certification Body <certificationBody>	A glossary of Certification Bodies is used to assign to Third Party Audits. Each record contains a Contact Name and supporting Email, Phone, Fax, and Address, which refers to an individual at a third-party organization. **Search Field:** Contact Name **Returned Fields:** Record Type; Contact Name; Email; Phone; Fax; Address; Certification Body record id

Record/ Element Name	Details
Supplier <supplier>	The Supplier record contains a Contact Name and associated Email, Phone, Fax, and Address fields, which refer to an individual who may or may not be a user of the system. **Search Field:** Contact Name **Returned Fields:** Record Type; Contact Name; Email; Phone; Fax; Address; Supplier record id; Supplier Name; Supplier Code
	The Supplier record also contains billing details entered using the registration wizard, including the name of an Individual or Department, a Billing Contact Name, and associated Email, Phone, Fax, and Address. They refer to an individual who may or may not be a user of the system. **Search Field:** Individual or Department; Billing Contact Name **Returned Fields:** Record Type; Individual or Department; Billing Contact Name; Email; Phone; Fax; Address; Supplier record id; Supplier Name; Supplier Code
Site <site>	The site may have a list of Growers associated to it for Produce type products. The Grower details contain a Grower Name and associated Email and Address, which refer to a third-party organization. **Search Field:** Grower Name **Returned Fields:** Record Type; Grower Name; Email; Address; Grower record id; Supplier Name; Supplier Code; Site Name; Site Code
Audit/Visit <audit>	The Audit/Visit record may have free text Name entries in the People Present table for attendees who are not users of the system. Entries for users also store the name in a *snapshot* text field. The Audit/Visit Issue record may have non-conformance actions with a free text Assigned To Name or a free text Completed By Name. Audit Checklists also share the Assigned To Name when a non-conformance is being created for a Checklist. **Search Field:** Name in People Present table; Issue Assigned To Name; Issue Completed By Name **Returned Fields:** Record Type; Name, Assigned To Name; Completed By Name; Audit/Visit record id; Audit/Visit Template Name; Audit/Visit Code; Supplier Name; Supplier Code; Site Name; Site Code; Audit/Visit Issue record id; Audit/Visit Issue Code

Record/ Element Name	Details
Product Specification <productSpecification>	The Product Specification record has a table of Printer Contact Name and associated Email, Phone, Fax, and Address, which refer to individuals at a third-party organization. **Search Field:** Printer Contact Name **Returned Fields:** Record Type; Printer Contact Name; Email; Phone; Fax; Address; Specification record id; Specification Number; Specification Version; Specification Name; Supplier Name; Supplier Code; Site Name; Site Code
	The FNF Product Specification record has a PIF Person Responsible and associated Email, Phone, and Fax, which refer to individuals at a third-party organization. **Search Field:** PIF Person Responsible **Returned Fields:** Record Type; PIF Person Responsible; Email; Phone; Fax; Specification record id; Specification Number; Specification Version; Specification Name; Supplier Name; Supplier Code
	The Product Specification has various approval names which are stored as free text. **Search Field:** Supplier Approved Name; Retailer Approved Name/Counter Ticket Issued By; Validation Override Approved By **Returned Fields:** Record Type; Supplier Approved Name, Retailer Approver Name; Validation Override Approved By; Specification record id; Specification Number; Specification Version; Specification Name; Supplier Name; Supplier Code
Product Record <productRecord>	The Product Record holds the name of the retailer approver of an associated specification/counter ticket. **Search Field:** Retailer Approved Name **Returned Fields:** Record Type; Retailer Approver Name; Product Record id; Product Number; Product Name; Supplier Name; Supplier Code

Security and Logging

The following Service and Endpoint configurations must be configured within the Brand Compliance Admin area, and assigned to the External Systems that are to have access to the Right to Access service:

Service:	DATAPRIV	Data Privacy Service
Endpoint:	DATAPRIV_ACCESS_POST	Creates a new DataPriv request and returns a link to the record's details
Endpoint:	DATAPRIV_ACCESS_GET	Retrieves DataPriv details for a given id

All calls to the service will be logged in the Brand Compliance Web Service Log.

The Data Privacy records created by the service are not viewable in Brand Compliance, and are not purged.

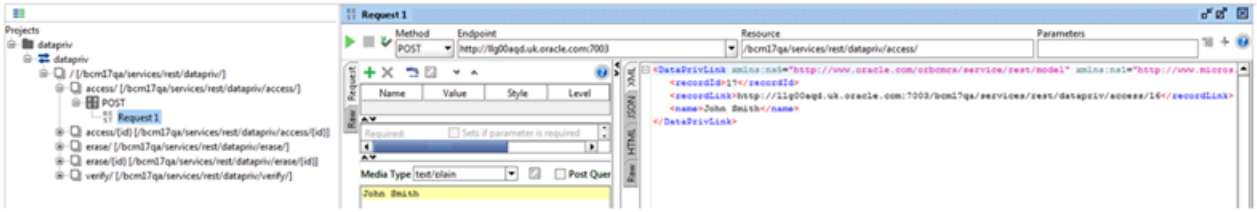
Example of Using the Right to Access Service

The request returns the actual data held, not the context in which it is used within the system. For example, to return the name and contact details of a Product Technologist,

but not the fact that within the system they are responsible for approving product specifications and so on.

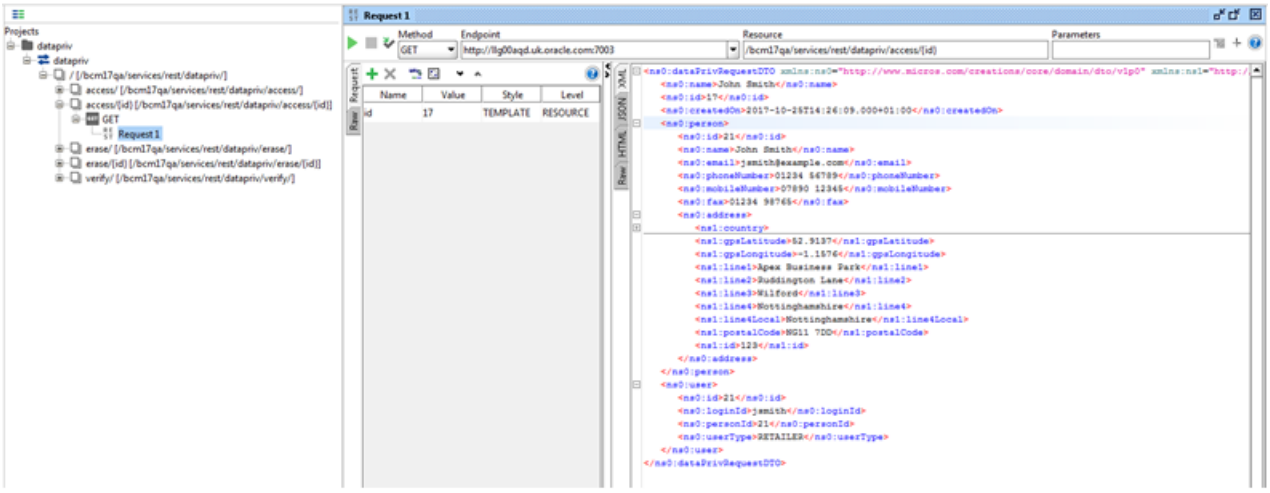
The stages involved in executing a Right to Access request are as follows. For this worked example, the call is made using a web service testing application.

1. Download the datatpriv WADL definition from the /services area of the Brand Compliance portal.
2. Submit a POST call to the services/rest/datapriv/access service, passing the name of the individual in the payload.



A recordLink element is returned with a link to a Data Privacy record.

3. Submit a GET call to the services/rest/datapriv/access service, passing the returned recordId value as the id parameter.



A message is returned containing the personal data that matches the name, within a dataprivRequestDTO root element. The structure of the XML message is as follows:

```
<ns0:dataprivRequestDTO xmlns:ns0="http://www.micros.com/creations/
http://www.micros.com/creations/core/domain/dto/v1p0/full" xmlns:ns
   <ns0:name>John Smith</ns0:name>
   <ns0:id>17</ns0:id>
   <ns0:createdOn>2017-09-07T09:01:50.000+01:00</ns0:createdOn>
   <ns0:person>
   <ns0:person>
   <ns0:user>
   <ns0:user>
   <ns0:designAgency>
   <ns0:counterTicketProducer>
   <ns0:certificationBody>
   <ns0:orderRequest>
   <ns0:retailer>
   <ns0:supplier>
   <ns0:supplier>
   <ns0:site>
   <ns0:contact>
   <ns0:contact>
   <ns0:audit>
   <ns0:audit>
   <ns0:audit>
   <ns0:productSpecification>
   <ns0:productSpecification>
   <ns0:productSpecification>
   <ns0:productSpecification>
   <ns0:productRecord>
   <ns0:productRecord>
</ns0:dataprivRequestDTO>
```

Returned Elements

Element Name	Details
name	The name of the individual as passed on the POST request.
id	The record id of the Data Privacy record created for this request.
createdOn	Timestamp of when the request was submitted.
person	Personal data fields from the Person records that match the name parameter.
user	Personal data fields from the User records that match the name parameter.
designAgency	Personal data fields from the Design Agency records that match the name parameter.
counterTicketProducer	Personal data fields from the Counter Ticket Producer records that match the name parameter.
certificationBody	Personal data fields from the Certification Body records that match the name parameter.
orderRequest	Personal data fields from the Order Request records that match the name parameter.
retailer	Personal data fields from the Company/Retailer record that matches the name parameter.
supplier	Personal data fields from the Supplier record that match the name parameter.
site	Personal data fields from the Site record that match the name parameter.
contact	Personal data fields from the Contact records that match the name parameter.
audit	Personal data fields from the Audit/Visit and Issue records that match the name parameter.
productSpecification	Personal data fields from the Product Specification records that match the name parameter.

Element Name	Details
productRecord	Personal data fields from the Product Records that match the name parameter.

The personal data fields are contained within tags that are similar to their (English) field labels in the UI:

```
<ns0:user>
   <ns0:id>7</ns0:id>
   <ns0:loginId>john smith</ns0:loginId>
   <ns0:name>John Smith</ns0:name>
   <ns0:email>jsmith-supplier@example.com</ns0:email>
   <ns0:userType>SUPPLIER</ns0:userType>
</ns0:user>
```

Additional metadata fields are included to identify the record by its code/id, name/description, and type.

Address elements may contain multiple localeData elements for country language translations:

```
<ns0:address>
         <ns1:country>
            <ns2:code>GB</ns2:code>
            <ns2:countryType>COUNTRY</ns2:countryType>
            <ns2:internetDomainSuffix>co.uk</ns2:internetDomainSuffix>
            <ns2:localeData>
               <ns1:description>Reino Unido</ns1:description>
               <ns1:locale>es</ns1:locale>
               <ns1:id>1904</ns1:id>
            </ns2:localeData>
```

There may be duplicates of the elements, where the system has more than one user with the same name, or where there are multiple records associated to an individual.

4. Edit the message using a text editor to remove duplicates, or entries that do not relate to the individual who made the request, and reformat if necessary, before delivering back to the requestor.

Right to be Forgotten Requests

The Right to be Forgotten service facilitates the request from the data subject (individual requestor) to the data controller (retailer/portal owner) to erase the personal data concerning them. The erasure of data is subject to certain predefined conditions within the system, and to the discretion of the data controller as to if/what data may be erased. The erasure is not reversible.

Data is erased by anonymizing it through obfuscation, that is, replacing the value with a string of random text.

When a retailer/portal owner receives a request, they use the *Right to be Forgotten* application they have developed to call the Data Privacy API. The API validates the request and erases the requested data from Brand Compliance. The retailer/portal owner then relays the result back to the requestor.

Personal Data Updated

Generally, personal data is erased from User and Contact records, with the anonymization of names and email addresses being reflected elsewhere in the system wherever they are referenced. Personal data that is not linked to a User or Contact record may also be erased; the retailer/portal owner must make a judgement on whether the data relates to the individual making the request.

The following table describes which areas of the system may have data erased.

Erasable Personal Data

Record	Fields
Users	Name; Email; Phone; Mobile; Fax; Login; Address
Contacts	Name; Email; Alternative Email; Phone; Mobile; Fax; Address
Company	Contact Name; Email; Phone; Fax
Order Request	Individual or Department; Contact Name; Email; Phone; Fax
Design Agency	Contact Name; Email; Phone; Fax; Address
Counter Ticket Producer	Contact Name; Email; Phone; Fax; Address
Certification Body	Contact Name; Email; Phone; Fax; Address
Supplier	Contact Name; Email; Phone; Fax Individual or Department; Billing Contact Name; Email; Phone; Fax in the Billing Details
Site	Grower Name; Email; Address in the Growers table
Audit/Visit	Name in the People Present table Assigned To Name; Completed By Name in the Issue record
Product Specification	Printer Contact Name; Email; Phone; Fax; Address in the OLC section's Printers table PIF Person Responsible; Email; Phone; Fax in the FNF specification's Formulation section Supplier Approved Name; Retailer Approver Name; Validation Override Approved By
Product Record	Retailer Approver Name

Text fields are replaced with randomly-generated text; the numeric GPS address fields are blanked; email addresses are set to x@example.com where x is 15 characters of randomly-generated text; countries and region selections are set to blank. If a field is blank, it remains unchanged.

If a User record is updated, the account is also deactivated and the password is expired.

Note: The anonymization of data in Brand Compliance as a result of a Right to be Forgotten erasure request is not reversible.

Personal data may also be updated using the individual APIs, or manually edited within the system. Physical deletion is not possible, due to referential integrity.

Exceptions

The key purpose of Brand Compliance is to assure due diligence in the management of the retailer/portal owner's products, and the suppliers thereof. There are legal or regulatory obligations to retain the approved technical product specifications for a set number of years, and during that time they may be required as legal evidence as part of a product safety issue. Where the approval or authorization by an individual forms part of the due diligence evidence, it is not feasible for the individual to be *forgotten*.

The following table indicates which fields are considered to be authorizations, and are preserved for due diligence purposes. These exceptions are therefore not erasable.

Preserved Personal Data

Record	Field
Scorecard	Completed By
Product Record	Retailer Approval
Produce Product Record	Retailer Approval
Temporary Specification	Approved By
Produce Specification	Approved By
	Counter Ticket Issued By
Product Specification	Supplier Approved Name
	Retailer Approved Name
	Validation Override Approved By
	Counter Ticket Issued By

Also, any names and email addresses within Pack Copy Files that have already been generated, and the Email, Batch Job, and Permissions Upload logs in the Admin area are preserved.

Purging Inactive User Accounts

The Right to be Forgotten service may also be used for purging user accounts that are no longer active, that is, those of users who have left the organization or are no longer involved in using the system. This is achieved by calling the API in the same way as if processing a Right to be Forgotten request.

Automatic date-based purging, such as a period of time since the user's account was disabled, or since their last login, is not provided, however the retailer/portal owner could feasibly develop a means of recursively calling the API based on a set of logic rules.

Security and Logging

The following Service and Endpoint configurations must be configured within the Brand Compliance Admin area, and assigned to the External Systems that are to have access to the Right to be Forgotten service:

Service:	DATAPRIV	Data Privacy Service
Endpoint:	DATAPRIV_VERIFY_POST	Creates a new verification request for a login id

Endpoint:	DATAPRIV_ERASE_POST	Creates a new Erasure request and returns a link to the record's details
Endpoint:	DATAPRIV_ERASE_GET	Retrieves Erase details for a given id

All calls to the service will be logged in the Brand Compliance Web Service Log.

The Data Privacy records created by the service are not viewable in Brand Compliance, and are not purged.

Entries within the Brand Compliance Change History logs, Attachments logs, and Comments logs use a foreign key to reference the name of the user who updated the record, attached the file, or added the comment. This means that if the name of the user is anonymized in the User record, the name will automatically become anonymized in the logs.

Updates to records to erase personal data through the Data Privacy API are not recorded in the record's Change History log. This avoids showing what the value was before and after anonymization.

Example of Using the Right to be Forgotten Service

The stages involved in executing a Right to be Forgotten request are as follows. For this worked example, the call is made using a web service testing application.

1. Download the datatpriv WADL definition from the /services area of the Brand Compliance portal.
2. Use the Right to Access service to retrieve the individual's set of personal data (see "Right to Access Requests").
3. Edit the returned XML message using a suitable text editor, removing duplicate user and person elements.

```
<ns0:dataPrivRequestDTO xmlns:ns0="http://www.micros.com/creation
http://www.micros.com/creations/core/domain/dto/v1p0/full" xmlns:
   <ns0:name>John Smith</ns0:name>
   <ns0:id>17</ns0:id>
   <ns0:createdOn>2017-10-25T14:26:09.000+01:00</ns0:createdOn>
   <ns0:person>
      <ns0:id>21</ns0:id>
      <ns0:name>John Smith</ns0:name>
      <ns0:email>jsmith@example.com</ns0:email>
      <ns0:phoneNumber>01234 56789</ns0:phoneNumber>
      <ns0:mobileNumber>07890 12345</ns0:mobileNumber>
      <ns0:fax>01234 98765</ns0:fax>
      <ns0:address>
   </ns0:person>
   <ns0:person>
      <ns0:id>7</ns0:id>
      <ns0:name>John Smith</ns0:name>
      <ns0:email>jsmith-supplier@example.com</ns0:email>
      <ns0:phoneNumber>09876 54321</ns0:phoneNumber>
      <ns0:mobileNumber>07890 12345</ns0:mobileNumber>
      <ns0:fax>09876 54320</ns0:fax>
      <ns0:address>
   </ns0:person>
   <ns0:user>
      <ns0:id>21</ns0:id>
      <ns0:loginId>jsmith</ns0:loginId>
      <ns0:personId>21</ns0:personId>
      <ns0:userType>RETAILER</ns0:userType>
   </ns0:user>
   <ns0:user>
      <ns0:id>7</ns0:id>
      <ns0:loginId>john smith</ns0:loginId>
      <ns0:name>John Smith</ns0:name>
      <ns0:email>jsmith-supplier@example.com</ns0:email>
      <ns0:userType>SUPPLIER</ns0:userType>
      <ns0:address>
         <ns1:id>36</ns1:id>
      </ns0:address>
   </ns0:user>
   <ns0:designAgency>
```

In this example, there are two pairs of user and person elements because there are two user accounts with the name John Smith - one for a retailer user and one for a

supplier user (user name and email address are not unique, it is the login id that uniquely identifies a user).

In order to pass validation, there must only be one user and person element in the submitted XML message. Remove any user and person elements that are not the individual being erased. Delete all the data between the opening and closing tags, just leaving a single person and user:

Note: The message is XML and can be edited using any UTF-8 compliant text editor.

It is vital that the structure and syntax of the message is maintained (with matching pairs of tags), so editing with a tool that recognizes the XML file format is recommended. For details of the message structure, see "Right to Access Requests".

Metadata fields that are included in the message to identify the records by their code/id, name/description or type are not erased. For details of which fields may be erased, see the "Erasable Personal Data" table.

Person/User/Contact Relationships

There may be a combination of user, person, and contact elements for an individual, depending on their role within the system:

- Each user of the system has a single User record and a single Person record.
- If a supplier user is a contact, in addition to their User and Person records, they have a Contact record for each of their designated contact roles.
- If an individual is a contact but they are not actually a user of the system, they have a Person record, plus a Contact record for each of their designated contact roles.

When submitting a Right to be Forgotten request, the following validation checks are applied:

- Only one user element must be present.
- Only one person must be present.
- The associated person element must be present for a user or a contact element.
- The personId in the user element must match the id in the person element.

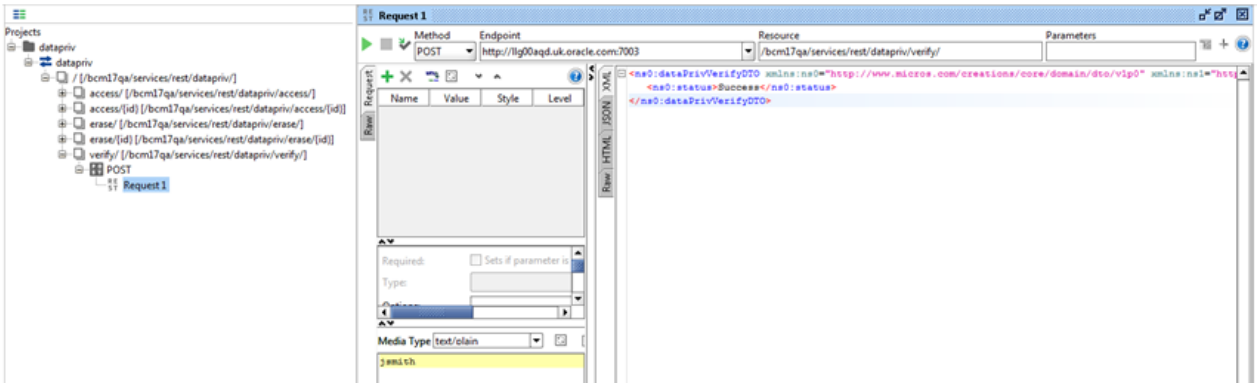
- The personId in the contact element must match the id in the person element.
- The user id in the contact element must match that in the user element.
- All contacts must be linked to the same person.
- A person element must have an associated user or contact element.

4. Make further edits to the XML message to check that the data in the elements for designAgency, supplier, productSpecification, and so on, all relate to the individual making the request. If not, delete the elements.

   When submitted, the personal data elements that remain in the message will result in the corresponding fields being erased in Brand Compliance (the field contents are replaced with anonymized text).

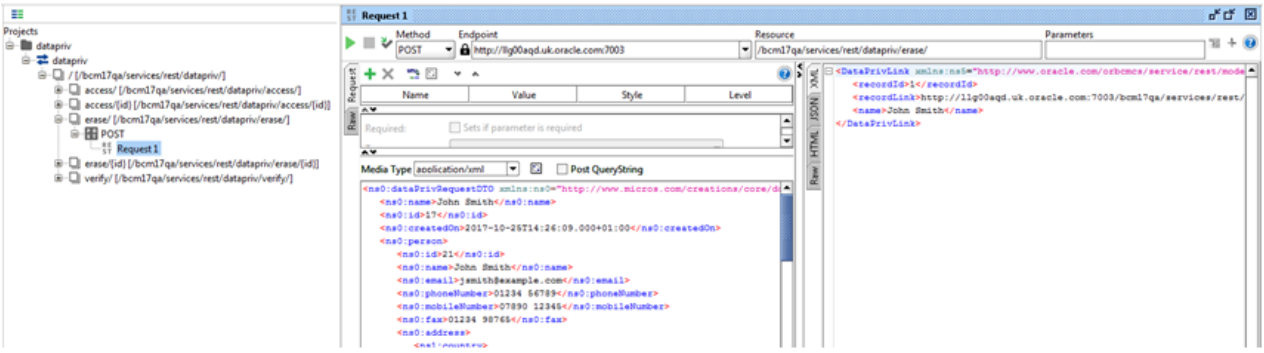
   If an element is removed, the field contents remain unchanged. For example, if wishing to erase the Grower Name but not the associated Email Address or Address within the Site element, delete the growerEmail and growerAddress elements from the XML message.

5. Optionally submit a POST call to the services/rest/datapriv/verify service, passing the login id of the individual (obtained from the User element of the XML message returned in Step 2) in the payload:



   This step checks whether the individual is responsible for the completion of a Scorecard, in which case they will not be permitted to be erased.

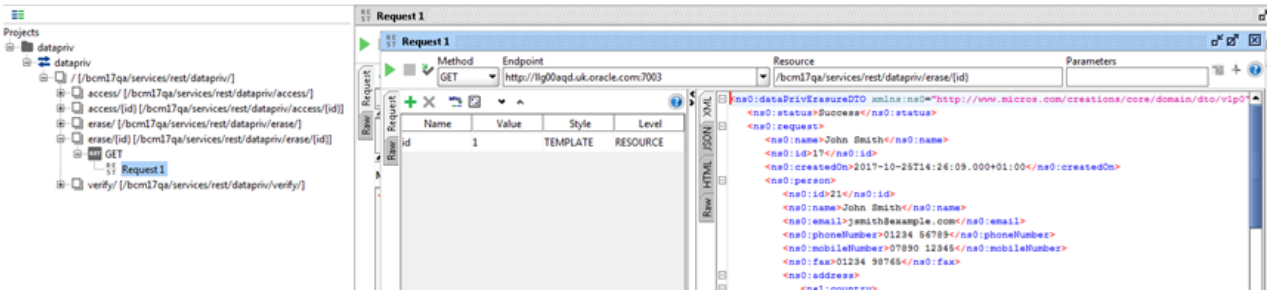
6. Submit a POST call to the services/rest/datapriv/erase service, passing the XML message in the payload:



   A recordLink element is returned with a link to a Data Privacy record.

   For possible failure conditions, see the "Error Messages" table.

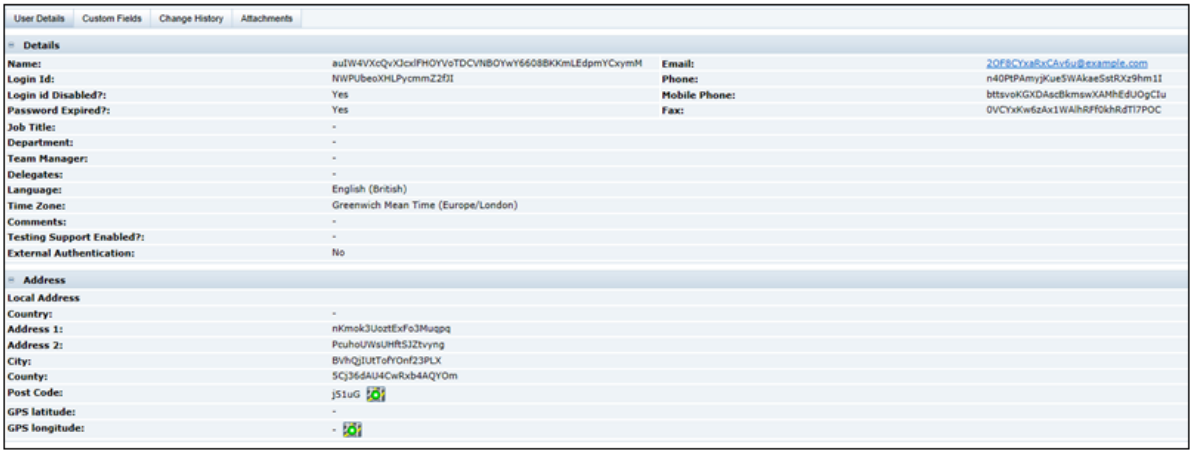
7. Submit a GET call to the services/rest/datapriv/erase service, passing the returned recordId value as the id parameter:



A confirmation message is returned, with the original data in a dataprivErasureDTO root element.

8. Optionally perform another a Right to Access request to check that the personal data has been erased as expected.

   Otherwise, viewing the erased records within Brand Compliance will show the obfuscated data:



9. Advise the requestor of the outcome of their request.

Other Aspects of Data Privacy

This section describes some other aspects of Data Privacy, and how they relate to Brand Compliance.

Data Portability

Data Portability is the right for the data subject to receive the data concerning them, in a common, structured machine-readable format. The Right to Access service and other APIs provide access to the majority of Brand Compliance data. The Reports module also allows the export of data in various formats.

Privacy by Design and Data Minimization

Privacy by Design and Data Minimization relates to making data privacy and protection integral to the design of the system, such as minimizing the data set to that which is necessary to perform the function of the system, and restricting the access to personal data to just the users that need it. The personal data held in Brand Compliance is already minimal; it relates to business contact data.

Access to the Brand Compliance data is restricted by the Permissions security model, based on the user's roles and what organization they are associated to; the configurable access controls are granular, down the level of individual fields.

There is no need to specifically mark fields as *sensitive* in Brand Compliance, however, the retailer/portal owner could chose to do so by configuring the field label system text themselves.

Data Deletion

The deletion of data is limited due to the due diligence nature of Brand Compliance. Certain data must be retained indefinitely, so options to delete or purge data are limited to a select few areas. Where data is deleted, it is not possible for it to be re-accessed.

The Right to be Forgotten solution includes the retailer/portal owner's ability to purge personal data from inactive user accounts. As certain data cannot be physically deleted, the erasure is by obfuscating/anonymizing the appropriate fields.

Notice and Consent

Notice and Consent is necessary where personal data is being held or processed. In Brand Compliance, the Terms and Conditions pages can be configured by the retailer/portal owner to include the necessary notices for the user to accept or otherwise.

Availability

Backup and disaster recovery facilities for Brand Compliance are provided as standard as part of the Cloud Service solution.

The majority of Brand Compliance data is available for export using the suite of secure APIs.

Tracking Technologies

Brand Compliance does not include any Tracking Technologies such as IP addresses or device fingerprints. A single cookie is used, for session connectivity.

Separation of Duties

Some tasks, such as the approval of a Product Specification or completion of an Audit, require actions by separate users; the completion of the tasks is controlled in Brand Compliance by workflow rules.

Logging

Maintenance of Brand Compliance records is recorded in the Change History log. Read access is not logged.

The Web Service log records details of all API calls; these are accessible only to administrators.

The Brand Compliance data is secured by HTTPS/SSL in transit; data at rest in the MySQL application database is not encrypted.