Web Development Programme

RESTful API Specification (Working Version)

Project	**Web Development Programme**
Prepared By	Richard Smith Development Lead (Explorer and API) Richard.M.Smith@ons.gov.uk
Reference	
Date	21/05/2010
Acknowledgement	

Document Control

Changes History

Issue No	Date	Change
0.1	20/01/2010	Initial version for comment to technical team only
0.2	29/01/2010	First complete version
0.3	12/02/2010	Updated following feedback from ONS and NOMIS
1.0	26/02/2010	Baselined
Working	11/05/2010	Document being kept up to date with changes but not reissued.

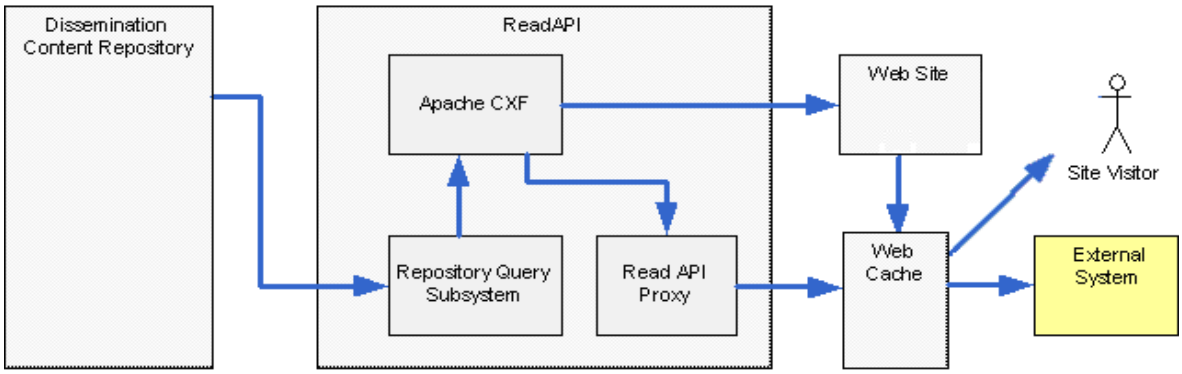
Table of Contents

1. Introduction

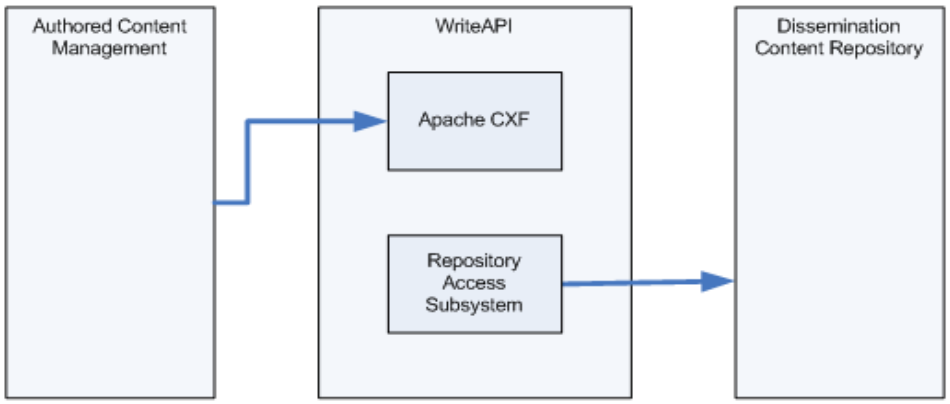
One of the key proposals from the Future of the Web initiative[11] (Appendix C) in 2008 was the use of an API to provide access to datasets, publications and other content items. This API would provide both read and write access to our datastores, for both internal and external users. It is intended that the Read API will be used as the foundation of many websites.

The high-level design for these APIs can be found in the WDP SAD[10] (the pictures below are from this). This document's purpose is to specify both the Read and Write APIs in detail, although this initial version is mainly the Read API. It is being circulated to Census and NOMIS as well as within WDP. Issues covered in this document were also discussed at the SDMX and Semantic Web workshop, and it was encouraging to discover that it is broadly consistent with the proposed standards for RESTful use of SDMX-based data.

1.1 Read API Diagram



1.2 Write API Diagram



2. Design Principles

Unlike the existing NeSS API[12] which uses the traditional SOAP / RPC approach, the WDP API will be implemented via the more modern RESTful approach as defined by Roy Fielding[2].

The main concept of a REST API is stateless interaction with a hierarchical structure of resources addressed using Uniform Resource Identifiers (URIs) and only the use of the verbs GET, PUT, POST and DELETE (and to a lesser extent TRACE, HEAD and OPTIONS).

This is harder to implement than a system based on procedure calls, but has advantages to the user such as not having to learn a different set of operations for each API they use.

In addition to the general principals of REST, the document Designing URI Sets for the Public Sector[1] provides further guidance detail around the structure to be implemented by public sector interfaces, and this is taken to be the foundation on which the API design is built. Much of this is intended to facilitate the use of RDF, but even though we won't be fully supporting RDF initially, it is a sensible framework to follow.

The service must also support the "discovery" of its own resources. To this end its responses will include references to related resources such that all resources can be discovered from the base URI entry point.

2.1 URI Types and Relationships

Although multiple URIs can reference the same resource, as stated in the W3C Documentation on Alaises[8], best practice is to maintain one 'real' URI per resource.

Designing URI Sets for the Public Sector[1] defines an optional schema for signifying URI type with the primary examples as follows:

- **Identifier**: "/id/…" designates an id URI identifying the non representational specific resource. This URI cannot return a resource directly but a 'document' URI can be resolved via a 303 response code.
  http://transport.data.gov.uk/id/road/M27/junction/9

- **Document**: a resolvable URI to a resource can be as the Identifier but with the 'id' segment removed ("/doc/…" can also be used but this could be seen as redundant text).
  http://transport.data.gov.uk/doc/road/M27/junction/9
  or http://transport.data.gov.uk/road/M27/junction/9

- **Representation**: as 'Document' but with file name and extension.
  http://transport.data.gov.uk/road/M27/junction/9/data.xml

- **Definition:** definition of a concept (or reference in our API)
  http://transport.data.gov.uk/def/road

- **List:** The same as document URI but with the Reference of the final concept/reference pair missing, in order to provide a list of URIs for that concept
  http://transport.data.gov.uk/road/M27/junction

So the identifier is a thing that exists, but is not an active internet resource. In RDF, these identifiers are used as the source and target in triples. The document is the actual internet resource corresponding to the identified thing, and the representation is the content presented in one of the formats available.

2.2 Resources with Hierarchical Structure

Some types of resource can themselves be broken down into sub resources in a hierarchical manner. In this situation the URI to the resource will return a representation of the complete resource while extending this URI will return a subset of the resource.

http://www.cars.com/make/alfaromeo (all alfa romeo cars stocked)
http://www.cars.com/make/alfaromeo/model/GT (all alfa GTs stocked)

The primary example for our API is that of a dataset. Extending the URI of the dataset by adding dimensions will address a section of the dataset or even a single observation. This allows all observations to be individually addressed.

http://www.statistics.gov.uk/data/contenttype/dataset/dsid/522.sdmx.compact.xml
(complete dataset as sdmx)
http://www.statistics.gov.uk/data/contenttype/dataset/dsid/522/sex/S2/age/A1.json
(single cell as JSON)

Cross-sectional data slices are best obtained via the query string rather than the main URI.

http://www.statistics.gov.uk/data/contenttype/dataset/dsid/522.sdmx.compact.xml?sex=S2&age=A1,A2,A3 (slice as sdmx)

2.3 Read - Write API Relationship

The presence of a resource makes it available to the read API (e.g. via GET). The type of resource, context of resource, and the identification of metadata fields, defines how the resource can be queried and discovered. The Write API defines these features therefore the Read API is created/maintained by the use of the Write API. The two are hence tightly associated.

2.4 Query Parameters and Search Integration

A URI request can match exactly to a resource…
http://www.statistics.gov.uk/data/contenttype/publication/pubid/3456
(give me publication 3456)

….or, if not, return a list of all child resources of that context (discoverability).
http://www.statistics.gov.uk/data/contenttype/publication/pubid
(give me a list of publications)

The returned 'list' could actually be a grid or other structure depending on the chosen representation (HTML, JSON etc.).

In this case URI parameters provide the ability for filtering of the returned list. Parameter/value pairs identify the resource field (e.g. body, title, named dimension) and associated query term.

http://www.statistics.gov.uk/data/contenttype/publication/pubid.html?title=Health And Efficiency

There are two types of querying possible when including parameters with the URI: exact match filtering and free text match filtering. Free text matching requires the use of a search engine and appropriate field indexing on write of the resource (Write API). Lucene will be used as the search engine using a similar architecture to the Publication Hub.

http://www.statistics.gov.uk/data/contenttype/publication/pubid?requiredFilterTerms/theme=Health

2.5 Connectedness

In a well-connected service, the client can make a path through the application by following links and filling out forms. In a service that's not connected, the client must use pre-defined rules to construct every URI it wants to visit. Right now the human web is very well-connected, because most pages on a web site can be reached by following links from the main page.

This API will attempt to guide the client from one application state to another. Discovery responses should enable the client to construct further discovery requests, and/or delivery resources.

2.6 PUT, POST and DELETE

2.6.1 PUT v POST

New resources can potentially be created via the Write API through PUT, POST or both. Initially the suggestion was that all new content be created via a PUT request, giving the calling application full control.

However, there are cases when we will want the Write API to make certain decisions, and therefore a POST operation will be used. The Write API will use the available metadata items for each resource in order to construct its URI, and write its resource record.

2.6.2 Safety and Idempotence

A GET or HEAD request cannot change anything and it should be safe to call it as many times as required, and it should be safe to send such requests to an unknown URI. It is not impossible for there to be side effects (such as hit counters) but we are not proposing any for our API.

PUT and DELETE operations will be idempotent, this implying that they can be repeated *ad nauseum*. New content can be PUT and if it is PUT again there will be no change of state. Similarly removing content via a DELETE, then removing it again has no effect, it is still gone and there is no change of state. Our API will obey these principles.

POST requests are neither safe not idempotent, intrinsically. The Write API must ensure there are no adverse effects.

2.6.3 Content Negotiation

The Roy Fielding query content negotiation discussion thread[4] explains how, according to Fielding, one should "... avoid content negotiation [a mechanism that allows several versions of a resource to be served by the same URI] without redirects like the plague because of its effect on caching". The point being conveyed is that multiple representations of a resource should have their own URI not be selected and returned for the same URI request based on negotiation headers.

However a representation URI for the resource can be negotiated via a 303 redirect to the document URI.

2.7 Asynchronous Use

SOAP services support asynchronous calls, whilst REST services traditionally don't. However this is no longer the case. A recent blog[17] describes how Apache CXF supports "continuations". This allows a percent complete to be repeatedly returned - ideal for an AJAX application.

3. Read API (Public)

3.1 ABNF Definition

ABNF[13] is a method of concisely defining the URI structure This is also the method used to define 'URI' by RFC 3986[14] which this is extending. Below is the first draft of the Read API's ABNF definition.

```
READAPIURI  =  "http://" domain context [version] pathToRes [pathInRes] [repName] [repType] ["?"
query]

domain      =  host              ; e.g. www.statistics.gov.uk
context  =  "/" 1*pchar          ; e.g. /onsapi
version  =  "/v" 1*DIGIT            ; e.g. v01
pathToRes   =  1*("/" concept ["/" reference])  ; using discovery metadata
pathInRes   =  1*("/" concept ["/" reference])  ; using dataset dimensions
~~repName =  1*("/" 1*ALPHA)          ; optional representation e.g. doc, def~~
repType  =  1*("." 1*ALPHA)      ; representation extension  e.g. .cross.sdmx.xml
query    =  queryParam *("&" queryParam)
queryParam  =  pagingParam / structParam / fieldParam
pagingParam =  ("noOfPages" / "noOfRows" / "noOfCols") "=" 1*DIGIT
pagingParam =/ ("pageOffset" / "rowOffset" / "columnOffset") "=" 1*DIGIT
structParam =  ("page" / "col" / "row") "=" field *("," field); attachment definition needed ?
fieldParam  =  queryType "/" field "=" refTerm *("," refTerm)  ; metadata = list of values
queryType   =  ("reft" / "prft" / "raft")            ; required, prohibited or ranking
field    =  (ALPHA *(DIGIT / ALPHA)) / "body"   ; TODO more characters are allowed
concept  =  (ALPHA *(DIGIT / ALPHA))        ; metadata or dimension name - see
reference   =  (ALPHA *(DIGIT / ALPHA))        ; metadata or dimension value - see
refTerm  =  *pchar               ; a search term to match against 'reference's
```

Special characters will need to be URL Encoded (e.g. space is %20)

3.2 Structure of URI

3.2.1 Domain / Context

For an internal API the domain name is relatively unimportant but should be selected to minimise likelihood of change. An external facing API must conform to the business provided domain. However the part of the domain designating the REST API context will be defined by either a sub domain or a context root.

e.g. http://www.statistics.gov.uk/data
or http://www.data.statistics.gov.uk
or http://www.statistics.data.gov.uk if going via the cross government hub[1]

This will be configurable by the system.

3.2.2 API Version Specifier (version)

The option for versions of the API to exist side by side can be achieved using the method employed by NOMIS[7]. Any version other than the current published would have a version specifier at this level, for example "…/v12/…". It should be remembered that '.' should be avoided in this specifier.

http://www.statistics.gov.uk/data/v03/

A version capability will be provided with each deployment of code such that a written resource will be available on all provided API versions. Configuration will define which version is used when no version specifier is included in the URI.

3.2.3 Path to Resource (pathToRes)

According to the W3C Opacity Axiom[6] a URI should be considered opaque by machine agents. That is associations to other URIs should not be inferred from the URI notation itself. Associated URIs should only be parsed from the *response* content. Roy Fielding explains in a discussion thread[5], how REST URIs are not *required* to be opaque and are encouraged to be human-meaningful, hierarchical identifiers.

This API will maintain the use of human readable hierarchical paths to resources, and within hierarchical resources (e.g. datasets). It will ensure all associations that must be parsed by systems are available from the URI response content ("hypermedia as the engine of application state"). The W3C URI Aliases documentation[8] recommends that a single URI only should be provided to the resource.

The hierarchical path segments will uniquely identify a resource and hence are made up from a subset of the metadata describing the resource. The decision on which metadata will be used should not be based on how an application web site will present the content (e.g. by taxonomy) but by unique identification with least potential to change over time. In other words the taxonomy *can* change without invalidating our URIs.

i.e.
http://www.statistics.gov.uk/data/contenttype/dataset/dsid.html?topic=unemployment
not http://www.statistics.gov.uk/data/contenttype/dataset/topic/unemployment.html
to get a list of datasets about unemployment (discovery)

and http://www.statistics.gov.uk/data/contenttype/dataset/dsid/un1.xml
not
http://www.statistics.gov.uk/data/contentType/dataset/topic/unemployment/dsid/un1.xml
to download one of the unemployment datasets with metadata and definitions (delivery)

Note that to retrieve part of what's held for a dataset (or other content item) an additional concept/reference pair is specified - this is contentpart/<name of content part>. For datasets the content parts will equate to SDMX artefacts. E.g

http://www.statistics.gov.uk/data/contenttype/dataset/dsid/un1/contentpart/data.xml
(just the data)
http://www.statistics.gov.uk/data/contenttype/dataset/dsid/un1/contentpart/definition.xml (just the DSD)

Often in blogging architecture, release date is used as once happened this can never change. Also content type (dataset, publication) can be used as changing this effectively changes the resource.

The flexibility to define the metadata to URI mapping is achieved by the Write API providing a content specification. This allows the above decision to be made on a content type by content type basis. The design hence allows the actual URI structure to be dictated by the client using the Write API.

3.2.4 Path within a Hierarchical Resource (pathInRes)

Designing URI Sets for the Public Sector[1] defines how a hierarchical resource, such as a dataset, can be internally addressed using the concept/reference URI segment pairs.

Examples of concept/reference pairs:

```
weather/snow
sport/bog-snorkelling
```

This allows a set of dimension and dimension value pairs to select subsets and observations with in the dataset.

Here is an example of the beta NOMIS API[7] following this structure.

https://nmtest.dur.ac.uk/websvc/SDMXService/query/NM_1_1/GEOGRAPHY/2092957699,2092957700,2092957701/SEX/5,6/ITEM/1/MEASURES/20100/FREQ/A/TIME/2004,2006/download/generic?signature=NPK-48e2faf3940ab84ee4bb0d:0xcbc1bd08750f1f115b8966fe6d83b6323989674c

However there are two issues with their method. First, the W3C URI Aliases documentation[8] defines that only one URI should identify a resource while the NOMIS API allows interchanging of the order of concepts; concept order should be fixed. Secondly NOMIS allows lists of references within each segment effectively converting the URI into a disguised query and no longer an identifier (described in another Roy Fielding discussion thread[3]).

This API will define the concept/reference pairs, and pair order, during the resource create operation (Write API) from the resource definition. For datasets this will be SDMX DSD concepts and dimensions in concept order.

A URI ending in a concept/reference will return the subsection of the dataset identified by the existing concept reference pairs. When ending with a concept only (*list* URI type) it will provide a list of links to available references for that context (possibly a subset of the associated code list) and all child contexts.

This allows requesting the information to drive a dimension explorer/picker being the actual available values. The DSD and code lists can also be queried however these may be much bigger than the values actually used by the dataset.

A full example of this is given in section 6.

This hierarchy provides a simple dataset subset capability through each dimension down to every observation but is limited due to the fixed ordering of concepts. This concept order can be defined for most likely use cases, for example geographical area coming first. More flexible subsetting is achieved via a query, described later.

3.2.5 URI Type Specifier (repName)

The redundancy of the specifier at the start of the URI means our API will not use it there, though we did consider using it placed at the end of the resource path, but this usage has now been dropped.

There are other specifiers according to Designing URI Sets for the Public Sector[1] which could be added if and when the API moves to a full RDF implementation. By following this approach we can fully support RDF in future if we decide to do so, including returning RDF as one of the supported output formats.

3.2.6 Representation Specifier (repType)

The Roy Fielding query parameters discussion thread[3] explains how, according to Fielding, the URL should include the representation type rather than it being a query parameter (due to the expectation of parameters made by intermediate systems). To separate this notation from URL hierarchy notation (forward slash) the preference is to use dot notation.

i.e. http://www.statistics.gov.uk/data/contenttype/dataset/dsid/un1.xml

not http://www.statistics.gov.uk/data/contenttype/dataset/dsid/un1?format=xml

Multiple dots are required where there are sub-options, such as sdmx.generic, sdmx.compact, sdmx.crosssectional.

The mapping of a *representational* URI to a *document* URI and to back end transformation methods will be managed through the Apache CXF framework.

Supported representations will be

```
.xml, .sdmx.compact, .sdmx.generic, .sdmx.crossectional
.html
.json
.nlm
.csv
```

Not every representation will be available for every URI. For more details see section 3.8.3.

3.2.7 Resource and Resource Type Association

The Write API will associate a resource type specification with every resource. This means the content type definition will be a resource in itself and navigable from the resource. A dataset will have a data structure definition along with associated code lists and associated reference and discovery metadata. XML content items will also have a content type definition and reference and discovery metadata.
These definitions can be discovered / browsed to generate GUIs for content selection.

```
Node: /data/contentType/dataset/dsid/uv09
Content Type: dataset
Title: Population by Ethnic Group
Primary Theme: Census
Dimensions:  ethnic group, geography
Release date: 20070101
Geographic: England and Wales
Codelists:   /data/contentType/dataset/uv09/contentpart/codelist/clid/CL_EthnicGrp1
             /data/contentType/dataset/uv09/contentpart/codelist/clid/CL_GeoPolicy
etc.
```

Note that the attributes which comprise resource type specification for each content type are derived from multiple entities and attributes within the WDP Logical Data Model[21]

3.2.8 Query Specification

As described in the previous section parameters should only be used for the purpose expected by intermediate systems i.e. defining a query. Also the URI path should be consistent with its purpose of uniquely identifying the resource. This leads to the decision that the query specification should be kept between the '?' and the end of the URI as name=value pairs separated by '&'.

A query will consist of field parameters and pagination parameters.

Section 3.3 defines the details of discovering and constructing query name/value pairs and how their processing is defined by the Write API.

It will also be possible to POST a query represented as XML in place of the use of GET parameters.

3.3 Search and Queries

As described previously standard URL parameters are used to convert a resource URL into a query URL. The parameters are a list of fieldname/value pairs between the first '?' and the

end of the URL. The parameters are separated by '&'. The parameters apply filtering to the resource context identified by the body of the URL.

A simple example is getting a list of datasets. A user of the Write API could determine that the release date is part of the URI, but let's say that it won't be, so it is necessary to query the release date.

http://www.statistics.gov.uk/data/contenttype/dataset/dsid?releaseDateFrom=20090101&releaseDateTo=20091231

The results of the query depends on the representation part of the URL (extension) and returns a representation in the same way as would have been without the parameters but as a limited subset. This may be a shortened list of matching content items associated to the URL context or a subset of a hierarchical resource such as a dataset.

Also pagination parameters are available which filters the list purely on offset and count. For structured responses pagination parameters will be available for up to three axes (e.g. row, column and page, or group, section and observation). For more details see section 3.4.

3.3.1 Field Definitions

A parameter name/value pair allows the application of a query string to a document field. Each resource has a set of defined field names to which a value is associated. The possible field names and values are identified by the Write API on creation of the resource, based on an associated content definition. For example content items matching a known metadata schema have their metadata name and values extracted as queriable fields. In addition there is always a 'body' field which contains any extracted text from the content (e.g. the words in a PDF). For datasets, dimensions can also be termed fields.

http://www.statistics.gov.uk/data/contenttype/dataset/dsid/u654.xml?geography=00MS&sex=S2

Each field type is defined by the content type definition which also defines properties for the field. This includes whether the field is free text or exact match queriable.

3.3.2 Query term syntax

3.3.2.1 Resource Queries

The requirements for the query terms are to allow exact match or text search comparison against chosen fields and the ability to apply AND, OR and NOT logic.

The structure of the query will reuse the method employed by the Publication Hub framework. This specifies an XML query schema. Such a query document could be POSTed to the URL. In addition GET parameters can perform the same function by using XPath style notation. Applications calling the API can freely choose to use GET or POST as they see fit with identical results. Where large and complex queries are involved, the use of POST will avoid potential issues with URL length.

The XML schema hence defines which parameters can be used. See Java Presentation User Guide[9] section 4 for the syntax. Here is an example:

```
<requiredFilterTerms>
   <rd_signoffStatus>(planned "provisional date visible")</rd_signoffStatus>
   <finalisedDate>
      <upper>
         <day>72</day>
         <month>0</month>
         <year>0</year>
```

```
      </upper>
   </finalisedDate>
</requiredFilterTerms>
<prohibitedFilterTerms>
   <market-sensitive>yes</market-sensitive>
</prohibitedFilterTerms>
<pageSize>500</pageSize>
<offset>0</offset>
```

Here is the GET parameter equivalent:

```
?requiredFilterTerms/rd_signoffStatus=(planned "provisional date visible")&
requiredFilterTerms/finalisedDate/upper/day=72&
requiredFilterTerms/finalisedDate/upper/month=0&
requiredFilterTerms/finalisedDate/upper/year=0&
prohibitedFilterTerms/market-sensitive=yes&
pageSize=500&
offset=0
```

The above states: get items with 'rd_signoffStatus' set as 'planned' OR 'provisional date visible', and 'finalisedDate' is less than 72 days from today, and 'market-sensitive' is not 'yes'. Return the first 500.
The fixed tag names will be configurable and will have abbreviated aliases (e.g. `reft/finalisedDate/u/d=72`).
There is also a 'rankingFilterTerms' which does not affect filtering but boosts matches when ordering by relevance.

3.3.2.2 Hierarchical Resource Filtering (Datasets)

The same style can be applied internal to datasets where fields map to dimensions:
XML query:

```
<reft>
   <age>a1,a2</age>
   <area>35</area>
   <date>
      <l>2001</l>
      <u>2006</u>
   </date>
</reft>
```

Parameter query:

```
contenttype/dataset/dsid/522?reft/age=a1,a2&reft/area=35&reft/date/l=2001&reft/date/u=
2006
```

The previous query returns a slice of dataset 522, however it would be possible to combine both types of query:

```
<reft>
   <theme>Economy</theme>
   <dimensions>age</dimensions>
   <age>a1,a2</age>
   <area>35</area>
</reft>
<prft>
   <designation>Experimental Official Statistics</designation>
</prft>
```

Parameter query:

```
contenttype/dataset?reft/theme=Economy&reft/dimensions=age&reft/age=a1,a2&reft/area=35
&pft/designation=Experimental+Official+Statistics
```

Return dataset slice a1, a2 and area 35 for all datasets that contain an age dimension and exist under the Economy theme. Exclude datasets designated "Experimental Official Statistics".

3.3.2.3 Handling of Time

Content items of all types including datasets have publication dates which is not the same as the time dimension. A time-series dataset may be republished every year or month, each time containing an additional time slice.

Thus a user wanting the latest time slice only cannot use the publication date. NDE uses a convention whereby if no date is specified, the latest is assumed - this suits the structure of the NeSS database where time series are held as separate dataset instances (rather than a growing cube). In the WDP API, any dimension which is not mentioned in the query is assumed to be "wildcarded" i.e. an implicit selection of all the dataset items. Therefore we need a convention and it will be the keyword "latest" i.e. `reft/time=latest`

3.3.2.4 Handling of Geography

Geographic hierarchies have always been a big issue for NeSS and NOMIS.

Unlike NOMIS, NeSS uses separate hierarchy entities such as "Admin 2005" and "Health 2003" which means each area has a single set of "children". This in theory should make shorthand notations to ask for "all the wards areas in Fareham LA" easier, but the hierarchy separation complicates it, so you have to say "all the wards in Fareham LA in the Admin 2005 hierarchy". This is achieved by using unique internal ids to select each area, rather than the SNAC code. Areas in different hierarchies with the same boundaries have the same SNAC code.

In NeSS Data Exchange you can discover the areaid list and use that to get the data

http://www.neighbourhood.statistics.gov.uk/NDE2/Disco/GetAreaAtLevel?LevelTypeId=140&AreaId=276980

http://www.neighbourhood.statistics.gov.uk/NDE2/Deli/getTables?Areas=281901,281902,281903,281904,281905,281906,281907,281908,281909,281910,281911,281912,281913&Datasets=67

In this case, a convenience method is also available

http://www.neighbourhood.statistics.gov.uk/NDE2/Deli/getChildAreaTables?ParentAreaId=276980&LevelTypeId=140&Datasets=67

For the WDP API, we want areas to be referenced exclusively by the new ids[22] which would be used in all system - an important linked data principle. One significant problem is that old areas will probably not be given a new id. If ONS generates these we're back where we were with SNAC codes (will the Change History Database help?)

Getting children is a geographic discovery function that can easily be satisfied via SMDX codelists using the "parentcode" attribute. The CORD system, which will underlie the WDP API has the ability to define "groups" of codes. So you could create one called 24UE_Wards. There would need to be some way to discover these, and this could be via parent area's definition - i.e. it would contain a list of all supported groups. So far so good.

But what about more advanced discovery questions like "what LA is postcode PO155RR in"? What area types do we hold data for for this dataset? What area types are parents or subordinates of another?

The answer is to extend the WDP logical data model to include geographic relationships similar to those held on NeSS. Physically, the content repository will hold a single entry for each area (and postcode). The SDMX codelists will reference this

http://www.statistics.gov.uk/data/contenttype/area/aid/PO155RR.html

would give you lots of information about the postcode, including its centroid and all the admin areas into which it falls.

http://www.statistics.gov.uk/data/contenttype/area/aid/24UE.html

gives all the information about Fareham include its children, grandchildren, parents, bounding box etc.

http://www.statistics.gov.uk/data/data/contenttype/dsid/uv09/def/codelist/clid/CL_GeoPolicy contains 24UE but not PO155RR. The code 24UE does exist as a resource in its own right, but this is a separate entity to the area itself, which is specified in the code's urn attribtute.

http://www.statistics.gov.uk/data/data/contenttype/dsid/uv09/def/codelist/clid/CL_GeoPolicy/codeid/24UE

```
<structure:Code value="24UE" parentCode="11"
urn="http://www.statistics.gov.uk/data/contentType/area/aid/24UE.html">
          <structure:Description xml:lang="en">Fareham</structure:Description>
</structure:Code>
```

Partial are name searches like this one in NDE
http://www.neighbourhood.statistics.gov.uk/NDE2/Disco/FindAreas?LevelTypeId=14&HierarchyId=0&AreaNamePart=Pear

would be achievable because when the Write API ingests content, it adds fields to the Lucene search index. This gives us the added value of search engine "tricks" such as misspellings.

3.4 Pagination and Structure

Pagination of the response is a useful additional feature which was tried out as part of the performance test demo.

pageoffset - start at page a
numberofpages - show b pages
rowoffset - start at row c
numberofrows - show d rows
coloffset - start at column e
numberofcols - show e columns

This is useful for both giving the user a convenient chunk for on-screen display (making page navigation easy) and for performance reasons (to limit the size of slices served in a single request). We may decide to make the default a number (e.g. 100) rather than everything, in which case a convention of -1 would be used to ask for all rows / cols.

We also have structure parameters which give the user control over which dimensions of a table go where.

row - list of dimension in row (nested) (Observation in SDMX)

col - list of dimensions in column (nested) (Section in SDMX)

page - list of dimensions in page (also called "wafer") (Group in SDMX)

If the user does not supply these parameters, defaults will be applied using the SDMX DSD (or it's equivalent on the database)

Example:

&row=Area&col=Age,Sex&page=Time

2005	Under 40		Over 40	
	Male	Female	Male	Female
Fareham	123	456	789	123

Gosport	456	789	123	456
New Forest	789	123	456	789

2006	Under 40		Over 40	
	Male	Female	Male	Female
Fareham	123	456	789	123
Gosport	456	789	123	456
New Forest	789	123	456	789

Etc.

Dimensions which are not named as “col” or “row” are assumed to be “page”. In a dataset with a large numbers of dimensions we would want to hide unreferenced dimensions (an implicit selection “All” or “Total”). This may not always be possible - if no total item exists would have to nest the variable in the heading.

3.5 Headers

Both Request and Response headers are useful in REST web services[19]. It is envisaged that the API will make use of the content-type parameter and also use the header to control caching, but this feature will not be in the initial version.

3.6 Security

3.6.1 Authentication

The cheapest, simplest and most user-friendly option is to have no authentication at all. This should be acceptable as we have separate instances of the Read API for the preview and live content, so everything in the live API should be published and public. The only real security issue is preventing premature release and this should hopefully be achieved via a mechanism that does not affect the public Read API.

Note that the new version of NDE has the login (username and password) and SSL removed by popular demand. At the recent SDMX and Semantic web workshop, there was a strong preference for an unauthenticated and unrestricted service.

On the other hand, Census anticipates a large increase in users from, possibly as early as mid 2011, and certainly late 2012. They would prefer to engage with distributors in advance and agree the levels of usage. These “distributors” are the heavier users (remember the API has many customers) and if limits (see also 3.6.2) are to be enforced then authentication is required. There is also a possibility of a chargeable version of the server with a higher limit than the free one.

At this point, no decision has been made, except that the first test release of the API will not feature any security measures. However, some possibilities are discussed below:

HTTP Basic and HTTP Digest are the two standard forms of authentication. HTTP Basic has limitations and HTTP Digest is more powerful though it does involve work for server and client, for example setting up “nonces”.

Apache CXF supports WS-Security[15] which includes the WSSE Username Token authentication method. This is what eBay does and it allows the credentials to be sent in a single URI such as
http://rest.api.ebay.com/restapi?CallName=GetSearchResults&RequestToken=UserToken&RequestUserId=UserName&Query=toy%20boat&Version=491&UnifiedInput=1
see Ebay ReST API[16] for more details.

Apache CXF also supports (via Spring) OPENID and OAUTH for security, which we intend to take a look at.

3.6.2 Licensing

The Read API Proxy component manages accesses and should be able to prevent overload of the servers. There is still a question over whether or not we want to limit usage by user. If so, an API Key like the eBay one above would be appropriate.

Cloud computing could present us with a method to support heavy usage. Ordnance Survey's OpenSpace API has usage limits but they are trialling the use of Amazon's cloud to support an unlimited "Pro" service.

Ideally we would not want to bother with anything more onerous than a simple click-use licence. Note that the existing 2001 licence is due for replacement by OPSI with the Creative Commons licence.

There are some drawbacks with the Creative Commons licence, such as issues with international usage, but it is probably our best bet. The Prioritised Requirements List for WDP includes the provision of an API key but this could be removed.

3.6.3 Protection

Even with public data, there is an issue with ensuring that it arrives safely without being tampered with. SSL (HTTPS) does add an overhead to the system, particularly if the response is verbose, for example NDE's GetSubjectTree operation runs 4 times faster without SSL. However, the integrity of our data is very important to ONS's reputation so we should consider encryption, provided performance is not too badly affected.

The recommendation is to test performance with and without SSL. A 128-bit SSL security certificate from Verisign costs about £1000 per server per year. Assume that the initial version will run over HTTP rather than HTTPS.

3.7 Notifications / Updates

Some kind of RSS or ATOM notification service is highly desirable, and is seen as a "must" by Census.

The RSS feed itself would be hosted by the ONS website rather than directly from the API itself. The RSS service would run a query using the API to detect publications and/or datasets released within a date range. Applications consuming the API can simply make a similar call, for example "give me all datasets published on or after 1st Feb 2010"

http://www.statistics.gov.uk/data/contenttype/dataset?reft/date/l=2010&reft/date/lower/y=2010&reft/date/lower/m=02/&reft/date/lower/d=01

A similar "pull" request is used by NDE via the operation getModifiedDatasets, though it does have the drawback that it cannot directly detect deletions. The handling of deletions in the new system is yet to be finalised at this point.

In addition to this, there is the possibility of expiration dates on data. So if an external system has downloaded a dataset, or a slice, it can note down when it should check for an update.

3.8 Response

3.8.1 Associations and Discoverability

Every level within a URI hierarchy must return a default representation that describes that context and all associations allowing a client to discover and navigate all associations, descendants and ancestors. In order to support both the download of a dataset and the discovery information from the same node, we use the contentpart concept identifier.

http://www.statistics.gov.uk/data/contenttype/dataset/dsid/uv09/contentpart/dimensions.json (gives a list of the dimensions in dataset UV09 as JSON)

http://www.statistics.gov.uk/data/contenttype/dataset/dsid/uv09/contentpart/data.compact.sdmx.xml (download UV09 as SDMX compact data, no definitions)

http://www.statistics.gov.uk/data/contentType/dataset/dsid/uv09.xml (the default - definition and data - format likely to be cross-sectional SDMX)

This allows a client to enter via the root entry point URI and find all resources down to dataset observations with no out-of-band information required (a REST principle). This can be done using HTML (using link tags), RDF or bespoke XML.

For example a request to a particular context, perhaps articles content types for May 2007, a list of all content items will be returned. If parameters are included, such as a filter on the taxonomy concept, then this list will be the matching subset.

Where free text query parameters have been included the response will also include sections containing:

- The queries used to generate the results.
- Pagination values.
- Field contents for each result for each field that was designated as 'stored' in the content definition (via Write API).
- URIs referencing suggestion alternate queries ("did you mean").
- Category lists with refinement counts for the appropriate metadata fields.

This is documented in the Java Presentation User Guide[9] section 4 (Appendix E).

Example (XHTML):

```
<table>
   <tr><th colspan='2'>Required Filter Terms</th></tr>
   <tr><td>theme</td><td>Census</td></tr>
   <td>topic</td><td>Ethnicity</td></tr>
</table>
<br/>
<table>
   <tr><th colspan='1'>Query Alternatives</th></tr>
   <tr><td>Did you mean "Electricity"?</td></tr>
</table>
<br/>
<table>
   <tr><th colspan='2'>Pagination Parameters</th></tr>
   <tr><td>pageoffset</td><td>0</td></tr>
   <tr><td>numberofpages</td><td>1</td></tr>
   <tr><td>rowoffset</td><td>0</td></tr>
   <tr><td>numberofrows</td><td>10</td></tr>
   <tr><td>coloffset</td><td>0</td></tr>
   <tr><td>numberofcols</td><td>10</td></tr>
</table>
<br/>
<table>
```

```
   <tr><th colspan='2'>Content Definition</th></tr>
   <tr><td>node</td><td>/data/contentType/dataset/dsid/uv09</td></tr>
   <tr><td>contenttype</td><td>dataset</td></tr>
   <tr><td>primarytheme</td><td>Census</td></tr>
   <tr><td>title</td><td>Population by Ethnic Group</td></tr>
   <tr><td>releasedate</td><td>20070101</td></tr>
   <tr><td>geographic</td><td>England and Wales</td></tr>
   <tr><td>dimension</td><td>ethnic group</td></tr>
   <tr><td>dimension</td><td>area</td></tr>
   <tr><td>codelist</td><td>/data/contenttype/dsid/uv09/contentpart/codelist/clid/CL_E
thnicGrp1</td></tr>
   <tr><td>codelist</td><td>/data/contenttype/dsid/uv09/contentpart/codelist/clid/CL_G
eoPolicy</td></tr>
</table>
<br/>
<table>
   <tr><th colspan='3'>Categories</th></tr>
   <tr><th colspan='1'>attribute</th><th colspan=1>coverage</th><th
colspan='1'>content-type</th></tr>
   <tr><td>name</td><td>England and Wales</td><td>Dataset</td></tr>
   <tr><td>value</td><td>England and Wales</td><td>Dataset</td></tr>
   <tr><td>rank</td><td>20</td><td>194</td></tr>
</table>
```

3.8.2 Hierarchical Resources

The default representation returned by a dataset concept / reference node when negotiated from the document URI should provide all necessary discoverable associations (e.g. next / previous concept and metadata). This can include references for all child concepts from this node down, not just those at the next level in the URI. This may depend on the response format - typically XML returning the whole tree and HTML just the immediate children.

Example: `Document for /data/contentType/dataset/dsid/uv09/ethnicgroup`

```
Parent:
Document for  /data/contentType/dsid/uv09

Children:
/data/contentType/dataset/dsid/uv09/ethnicgroup/white
/data/contentType/dataset/dsid/uv09/ethnicgroup/black
/data/contentType/dataset/dsid/uv09/ethnicgroup/asian
/data/contentType/dataset/dsid/uv09/ethnicgroup/chinese
/data/contentType/dataset/dsid/uv09/ethnicgroup/other

GrandChildren:
/data/contentType/dataset/dsid/uv09/ethnicgroup/white/geography/AA01
/data/contentType/dataset/dsid/uv09/ethnicgroup/white/geography/AA02
etc.
```

3.8.3 Representation

HTML is the easiest and most human friendly format for describing associations and can also double to return a hierarchical resource representation. At this stage resource descriptions shall be done as XHTML but will allow for the addition of RDF in the future.

The HTML format returned can be arranged in a normalised XHTML table structure. This structure can be determined by adhering to accessibility rules for tables in terms of row / column alignment. This satisfies usability as well as discoverability while providing an optional representation which can be used for transform to a presentational format (the XHTML table is not intended for direct presentation).

Where XML is requested as the output format, for datasets this will by default be SDMX where an appropriate artefact exists, for other discovery responses will be in XHTML. The

ONS SDMX Standards document[20] will be adhered to for the details of how data slices and definition artefacts are structured. If we adopt a single default data format (the ONS flavour of cross-sectional) there will be no need to specify this in the request, otherwise multiple dots will be used e.g. compact.sdmx.xml.

Because we are generating our SDMX output from CORD, rather than serving SDMX artefacts from a file system, we can extend the service to support other XML formats (such as LGDX) in future. The initial version of the API will be SDMX only.

JSON is a popular and useful return format and the API should support it. NLM too is another one the API should support if possible.

It is worth noting that some client platforms are capable of converting the response themselves, making some formats less important to directly supply. For example, XLS can be avoided. Excel 2007 can open XML and XHTML documents directly. CSV will have to be supported for downloads, and the API could optionally change the MIME type to fake an XLS file. Adobe Flex's httpService command will attempt to convert the response to an object, e4x, xml dom, json, html (and others) on request, but this is an extreme example.

3.8.4 Codes

The API will return standard HTTP status codes[18] [19]. The user documentation will include some information about the expected codes (e.g. when redirects are used).

3.9 NFRs

3.9.1 Response Times

The current NFR states that a 10,000 row dataset query should return a response to the edge of the ONS network within 2 seconds. Performance tests run using a test API and a 10,000,000 row census dataset were very encouraging, with data slices being returned typically in about 1-2 seconds. The handling of large requests is discussed below.

3.9.2 Size Limitations

When a user requests a complete dataset, or a large slice, there is a chance that the output will be too large to be sensibly returned in a single response. To deal with this, chunking via pagination parameters can be used. If the user does not specify any paging options, a default chunk size could be imposed (e.g. 100 rows x 100 cols x 10 pages = 100,000 cells). The value of this default could be tuned following performance tests. An alternative is the use of an asynchronous call (see section 2.7).

3.9.3 Availability

Unless otherwise advised, we'll assume the 99.7% uptime target for the web site will be shared by the API. There is an issue with whether or not we give an SLA or a target. Although customers may well appreciate a formal SLA, the expedient route (and the one used by NDE) is to provide a target which is monitored. Customers can be given confidence using the statistics relating to the achievement of this target, provided they are good!

The initial test version is being deployed to the NeSS PRS (external test) server, which is occasionally used for performance tests and so may be unavailable or slow from time to time. No such activities are planned during March and April.

4. Write API (ONS Internal Only)

In a RESTful system the URIs remain the same whether reading or writing therefore the URI discussion from the Read API applies here.
The act of writing a resource with its assigned content type definition brings its access via the Read API into existence and makes resource accessible in three ways:

- Fetchable – GET using its URI
- Discoverable – referenced by associations from URI parent contexts
- Queryable – referenced by associations from a parent context with matching parameter fields

The Write API is also used to register datasets, its sole function in Functional Release 1.

Further details of the Write API will be in a later version of this document.

5. Other Issues

5.1 Documentation

5.1.1 Read API

The Read API should ideally have lightweight web-based documentation with examples. A very simple demo application should also be supplied for external users (e.g. a web page using AJAX).

It should not be necessary to supply a WADL or v2.0 WSDL document. The ABNF specification is concise and useful and should be included.

5.1.2 Write API

The Write API will initially only be usable internally and in this case it is likely that a traditional user manual will be best.

6. Worked Example

6.1 Exploring Data with the Read API

6.1.1 Query

Our start point will be the search results for datasets matching a particular query.

e.g. http://www.statistics.gov.uk/data/contenttype/dataset/dsid?reft/theme=Population

The query returns a list of datasets as a grid with id, name, date and theme (loosely based on wireframe DT2)

	id	Name	Release Date	Theme
Explore Download	uv09	Ethnic Group	01/04/2001	Population
Explore Download	pproj1	Population Projections	01/01/2009	Population

If the user clicks on one of the "i" buttons the explorer will retrieve selected information about the dataset, most likely using more than one API call. For example one to get the dataset details (originally from the CMS)

http://www.statistics.gov.uk/data/contenttype/dataset/dsid/contentpart/datasetdetails.html

And another to get the SDMX DSD

http://www.statistics.gov.uk/data/contenttype/dataset/dsid/uv09/contentpart/definition.xml (dsd)

The output would also include links to further discoverable information, such as the codelists used by the dataset. E.g.

http://www.statistics.gov.uk/data/contenttype/data/dsid/uv09/contentpart/codelist/clid/CL_Ethnic_Grp.xml

6.1.2 Download

If the user clicks on Download, then he will be prompted for the required format (SDMX, CSV, XLS etc.). XML will be the default XML format (SDMX).

http://www.statistics.gov.uk/data/contenttype/dataset/dsid/uv09.sdmx.xml

http://www.statistics.gov.uk/data/contenttype/dataset/dsid/uv09.csv

For large datasets this may require the use of "continuations" and/or some kind of warning, where a cell limit threshold has been exceeded.

6.1.3 Explore

If the user clicks on Explore, he is presented with a list of dimensions to pick from. The dimension lists will be driven from the codelists associated with the dataset, which may already be a subset of the possible values. Proposal is that codelists under a dataset node will only contain values used in the dataset, "freestanding" codelists will have all values.

Even when using dataset-specific codelists to drive a dimension item picker there is still a chance of getting blank cells in the table.

To further minimise the chance of blanks, the API allows you to filter each picker based on previous selections (using the query string).

UV09 has dimensions Geography and Ethnic Group

Whichever dimension is picked first is unfiltered. Let's say we pick geography and we know the codelist is called cl_geog_ew_oa

http://www.statistics.gov.uk/data/contenttype/dataset/dsid/uv09/contentpart/codelist/clid/cl_geog_ew_oa

gives a list of all areas the dataset covers… or does it?

CORD has functions to generate a hierarchical response where needed, such as geography and SIC codes. The proposed logic is as follows: If the response is HTML, it is most likely the client will want just one level down, if XML (SDMX), the entire tree.

http://www.statistics.gov.uk/data/contenttype/dataset/dsid/uv09/contentpart/codelist/clid/cl_geog_ew_oa.html

returns England and Wales (code 727)

http://www.statistics.gov.uk/data/contenttype/dataset/dsid/uv09/contentpart/codelist/clid/cl_geog_ew_oa/codeid/727.html

returns England (064), Wales (220)

http://www.statistics.gov.uk/data/contenttype/dataset/dsid/uv09/contentpart/codelist/clid/cl_geog_ew_oa/codeid/064.html

returns North East, South East, South West etc.

For the next dimension a query is required (or accept the chance of blank cells)

http://www.statistics.gov.uk/data/contenttype/dataset/dsid/uv09/contentpart/codelist/clid/cl_ethnic_grp.html?reft/geography=064,220

This asks the service to filter the codelist to only those having data for England (country) and Wales (country).

We can now put together the "delivery" slice

http://www.statistics.gov.uk/data/contenttype/dataset/dsid/uv09.html?reft/geography=064,220&reft/ethnicgroup=EG1,EG2,EG3

In the above HTML has been requested (doc is optional)

http://www.statistics.gov.uk/data/contenttype/dataset/dsid/uv09.crosssectional.sdmx.xml?reft/geography=064,220&reft/ethnicgroup=EG1,EG2,EG3

This one asks for cross-sectional SDMX.

Appendix A References

1. Designing URI Sets for the UK Public Sector, version 1.0, October 2009 http://www.cabinetoffice.gov.uk/cio/chief_technology_officer/public_sector_ia.aspx
2. Roy T. Fielding dissertation on REST http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm
3. Roy Fielding discussion: query parameters http://tech.groups.yahoo.com/group/rest-discuss/message/11153
4. Roy Fielding discussion: content negotiation http://tech.groups.yahoo.com/group/rest-discuss/message/5857
5. Roy Fielding discussion: opaque URIs http://tech.groups.yahoo.com/group/rest-discuss/message/3232
6. Opacity axiom http://www.w3.org/DesignIssues/Axioms.html#opaque
7. Nomis REST service https://nmtest.dur.ac.uk/Default.asp instructions : Notes://TGROUP1/8025754500605EAF/DC7D63C5D4D13EEA002565830053C117/4D087AAEAF4871478025766600 5729C7
8. URI Aliases http://www.w3.org/TR/webarch/#uri-aliases
9. Java Presentation Framework User Guide Notes://TGROUP1/802571090060D02E/DC7D63C5D4D13EEA002565830053C117/622BEA5E863B27378025760E0059113C
10. Web Development Program SAD Notes://TGROUP1/802575B00042EC75/DC7D63C5D4D13EEA002565830053C117/468FC03C311D29B4802576870032DABF
11. Future of the Web Notes://TGROUP1/802575B00042EC75/DC7D63C5D4D13EEA002565830053C117/E191E6E1341508718025760F002AEBAB
12. NeSS Data Exchange http://neighbourhood.statistics.gov.uk/dissemination/Info.do?page=nde.htm
13. ABNF Definition http://en.wikipedia.org/wiki/ABNF
14. RFC 3986 http://tools.ietf.org/html/rfc3986#appendix-A
15. WS-Security in Apache CXF http://domagojtechtips.blogspot.com/2007/08/cxf-spring-and-ws-security-putting-it.html
16. Ebay REST API http://developer.ebay.com/developercenter/rest/ebayrestapiguide.pdf
17. Continuations in CXF http://sberyozkin.blogspot.com/2008/12/continuations-in-cxf.html
18. HTTP status code http://en.wikipedia.org/wiki/List_of_HTTP_status_codes
19. Developing a REST Web service (includes codes and headers info) http://www.packtpub.com/article/developing-rest-based-web-service
20. ONS SDMX Standards notes://TGROUP1/802575B00042EC75/DC7D63C5D4D13EEA002565830053C117/583E4121F99495AC80257600004D3B67
21. WDP Logical Data Model Notes://TGROUP1/802575B00042EC75/DC7D63C5D4D13EEA002565830053C117/5F36C2D02C219849802576B20046C521
22. OS Naming and Coding Policy http://www.ons.gov.uk/about-statistics/geography/policy/coding-and-naming-for-statistical-geographies/index.html

Appendix B Glossary

AJAX - Asynchronous Javascript and XML http://en.wikipedia.org/wiki/AJAX

ATOM - XML standard for web feeds http://en.wikipedia.org/wiki/ATOM

Apache CXF - Web service framework http://cxf.apache.org/

Click-use Licence http://www.opsi.gov.uk/click-use/index

Content Negotiation - http://en.wikipedia.org/wiki/Content_Negotiation

Content Specification - metadata held for a content item

Content Type Definition - template for what metadata is held by a content item

CORD - Central ONS Repository for Data - Database with associated functional layers designed to hold all ONS data, currently used only by National Accounts.

Creative Commons Licence http://en.wikipedia.org/wiki/Creative_Commons_licenses

CSV - Comma Separated Values http://en.wikipedia.org/wiki/Comma-separated_values

Discoverability - ability to discover new content
http://research.yahoo.com/files/discovery.pdf

HTML - Hypertext Markup Language http://en.wikipedia.org/wiki/HTML

HTTP (GET, PUT, POST DELETE) - Hypertext Transfer Protocol

http://en.wikipedia.org/wiki/HTTP

HTTP Basic - simple web authentication method
http://en.wikipedia.org/wiki/Basic_access_authentication

HTTP Digest - more advanced authentication method
http://en.wikipedia.org/wiki/Digest_access_authentication

Idempotence - http://en.wikipedia.org/wiki/HTTP

JSON - Java Script Object Notation http://en.wikipedia.org/wiki/JSON

LGDX - Local Government Data eXchange - XML format design by CLG for local data
http://www.neighbourhood.statistics.gov.uk/dissemination/Info.do?page=nde.htm

Lucene - Open source search engine http://lucene.apache.org/

NDE - Ness Data Exchange (web service)
http://neighbourhood.statistics.gov.uk/dissemination/Info.do?page=nde.htm

NeSS - Neighbourhood Statistics Service
http://neighbourhood.statistics.gov.uk/dissemination/Info.do?page=aboutneighbourhood/about.htm

NOMIS - https://www.nomisweb.co.uk/Default.asp

NLM - An XML format from the National Library of Medicine (originally intended for scientific publications) http://xml.coverpages.org/nlmXML.html

OAUTH - Open security protocol http://en.wikipedia.org/wiki/Oauth

OPSI - Office of Public Sector Information http://www.opsi.gov.uk/

Out-of-band - http://en.wikipedia.org/wiki/Out-of-band

OPENID - Open authentication standard http://en.wikipedia.org/wiki/Openid

OpenSpace - mapping API from Ordnance Survey
http://openspace.ordnancesurvey.co.uk/openspace/

Nonce - number used once - a cryptographic tool
http://en.wikipedia.org/wiki/Cryptographic_nonce

POX - Plain old XML (no schema) http://en.wikipedia.org/wiki/Plain_Old_XML

Publication Hub (or Pub Hub) - ONS web-based system for managing publications

Preview (repository) - version of the repository not available to the public

RDF - Resource Description Framework
http://en.wikipedia.org/wiki/Resource_Description_Framework

Resource Type Specification - template for information held for a particular resource type

REST - Representational State Transfer - http://en.wikipedia.org/wiki/REST

RPC - Remote Procedure Call http://en.wikipedia.org/wiki/Remote_procedure_call

RSS - Popular web feed format http://en.wikipedia.org/wiki/RSS

Safety - http://en.wikipedia.org/wiki/HTTP

SDMX - Statistical Data Metadata Exchange - http://sdmx.org/

SIC - Standard Industrial Classification
http://en.wikipedia.org/wiki/Standard_Industrial_Classification

SOAP - Simple Object Access Protocol http://en.wikipedia.org/wiki/SOAP_(protocol)

Spring - Development framework for Java and .Net
http://en.wikipedia.org/wiki/Spring_Framework

SSL - Secure Socket Layer http://en.wikipedia.org/wiki/Transport_Layer_Security

URI - Uniform Resource Identifier http://en.wikipedia.org/wiki/URI

URL - Uniform Resource Locator http://en.wikipedia.org/wiki/URL

WADL - http://en.wikipedia.org/wiki/Web_Application_Description_Language

WSDL - http://en.wikipedia.org/wiki/Web_Services_Description_Language

WS-Security - Web services security protocol (mainly for SOAP)
http://en.wikipedia.org/wiki/WS-Security

WSSE Username Token - Web services secure identifier http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0.pdf

XML - Extensible Markup Language http://en.wikipedia.org/wiki/XML

Appendix C Future of the Web Excerpt (from 2008)

Removed.

Appendix D ONS SDMX Standards Excerpt

Deciding on the type of Dataset

The next step is to determine the type of Dataset being defined, as this has a significant bearing on the placement of the Dimensions and Attributes in the Data Structure Definition.

There are three possible types of Dataset which are permitted within SDMX as deployed at ONS:

- Time Series
- Cross Sectional (with a single Cross Sectional Measure)
- Cross Sectional (with no Cross Sectional Measure)

For the remainder of this document a Cross Sectional Measure will be referred to as an XS Measure.

This is a relatively simple exercise for someone who is familiar with the Dataset and there are two simple questions to be answered;

- Is the data within the Dataset measured over multiple time periods? In which case the Dataset type is Time Series
- Is the data within the Dataset measured over a single time period? In which case the Dataset type may be either with or without an XS Measure

It is recommended that an XS Measure is defined, but if the Dataset really doesn't have a Dimension which could be used as the XS Measure this is allowed.

Defining the Data Structure

The structure of a Dataset is pretty much standard for all three types of Dataset. It must have the following levels defined;

- Group
- Series (Time Series) or Section (Cross Sectional)
- Measure
- Observation

As a general rule Attributes can be held (attached) at any level, the following describes the levels at which Dimensions must be held (attached).

In **ALL** cases the Frequency must be the first Dimension, as suggested in the SDMX Implementers Guide (page 26).

It is also mandated that Dimensions and Attributes are ordered in the DSD in the same sequence as the data, as this negates the need to create a mapping file for the SDMX converter.

Time Series

This is a Dataset measured over multiple periods of time.

At the Group level the Frequency of the Dataset, the frequency at which the data collected, must be defined.

At the Series level ALL of the dimensions except Time, must be defined.

At the Measure level the Time, together with the Observation value must be defined. In the resulting SDMX-ML file the Observation Value is held as an object named 'OBS_VALUE'.

Cross Sectional

This is a Dataset measured over a single period of time. Consequently, a Dimension other than Time must be used as the Measure.

The rules for defining a Cross Sectional Dataset are more relaxed than those for a Time Series Dataset.

Dimensions can be associated at the Group, Section and Measure levels.

As mentioned there are two potential types of Cross Sectional Dataset, one with an XS Measure and the other without an XS Measure.

With an XS Measure

At the Group level the Frequency of the Dataset, the Time (both of these are mandatory), together with any other Dimensions must be defined.

At the Section level ALL of the non-Measure dimensions must be defined.

At the Measure level the XS Measure Dimension must be defined.

At the Observation level the value of the Observation, the count or percentage etc must be defined. In the resulting SDMX-ML file the Observation Value is held as an object named 'OBS_VALUE'. Also held at this level are the SDMX attributes of OBS_STATUS and OBS_CONF.

Without an XS Measure

At the Group level the Frequency of the Dataset, the Time, only the Frequency is mandatory, together with any other Dimensions must be defined.

No Dimensions are defined at this level.

At the Measure level the Observation is defined together with any Dimensions.

At the Observation level the value of the Observation, the count or percentage etc must be defined. In the resulting SDMX-ML file the Observation Value is held as an object named 'OBS_VALUE'. Also held at this level are the SDMX attributes of OBS_STATUS and OBS_CONF.

Appendix E Read API Features for FR1

Ref	Function	FR1
1	Test data	✓
1	Real data	×
3.8.1	Basic content item metadata for datasets and other nodes	✓
3.8.1	DSD and Codelists	✓
3.8.1	MSDs and Metadata sets	×
3.5	Caching and headers	×
6.1	Simple navigation of geography (buy codelist)	✓
3.3.2.4	Advanced geographic facilities	×
3.6	Security / API key	×
3.8	Full dataset download (CSV and SDMX)	✓
3.7	RSS notifications	×
3.3	Get data slice by dimension values (in query string)	✓
3.3	Get data slice by advanced search	×
3.3	Get list of datasets by simple content definition queries	✓
3.3	Get list of datasets by advanced search	×
3.4	Pagination and row / column specification	✓
3.3.2	Tridion content (publications)	×
2.1	Support for RDF / Semantic web	×
2.7	Asynchronous calls / throttling / cloud	×
3.9.2	Size limits and default page size	×
3.8.3	Output format: CSV (datasets)	✓
3.8.3	Output format: XML (SDMX)	✓
3.8.3	Output format: XHTML (discovery)	✓
3.8.3	Output format: JSON	×
3.8.3	Output format: NLM (publications)	×
3.8.3	Output format: RDF	×
5.1	Basic instructions page (derived from spec)	✓
5.1	Fully signed off user documentation	×

Appendix F Constraints and Limitations

1. Number of Dimensions

The CORD database currently allows for a maximum of 10 dimensions. Three of these are taken up with the attributes Observation Status, Unit Multiplier and Unit which are treated as dimensions in CORD. So in effect only 7 dimensions will be supported, one of which is normally Date.

2. Number of Rows, Columns and Pages

In FR1, these are limited to 10000 each. This is an arbitrary figure.