



QuaLiFY

EuroFIR Web Services:

Specifications for EuroFIR request-response based Web services

Version 1.2

April 2015

SUMMARY

EuroFIR Web Services provide an interface for the food composition information data interchange. This specification defines the interface for these EuroFIR Web services and the format for the requests and responses. Moreover, the specification defines the authentication methods. The requests use specially designed *Food Data Query Language* (FDQL), which is part of the specification. The responses use *Food Data Transport Package* (FDTP) for delivering food composition information and specially designed *Meta Data Transport Package* (MDTP) for delivering meta information. The current version of the EuroFIR Web Services supports Simple Object Access (SOAP) protocol.

Content

Introduction.....	5
Objective	5
Background	5
Terminology.....	7
General Terms	8
Process Flow	9
Web Service Request.....	10
Web Service Response.....	11
Normal Response	11
Error Message	12
Advice for the developers	13
Web Services Authentication	14
Authentication keys.....	14
Signing the request.....	15
Checking the request.....	15
Risk management of the identification keys	15
Authentication errors.....	15
Advice for the developers	16
Food Data Query Language (FDQL).....	16
FDQL Data Model.....	16
FDQL Data Model Restrictions	19
FDQL Reserved Words	20
FDQL Implementation.....	24
FDQL Sentence Structure in XML	24
Example of the FDQL Sentence	25
Semantic Rules for the FDQL Sentence Translation	27
General Interpretation Rules.....	27
Joins Between the Entities.....	28
Validation of the FDQL Sentence	30
WHERE-clause Evaluation	31
Ordering with the ORDER BY -clause	32
Food related semantic rules	32
Component Related Rules.....	33
Component Value Related Rules	35
Metainformation Related Rules	36
Metadata Transport Package (MDTP)	38
Main structure of the MDTP	38
Grouping -structure	39
FCDB_Describe.....	40
ComponentList.....	40
FoodList.....	40
TermList.....	41
Grouping	41
Web Services	42
GetComponentList.....	42
GetContentInformation	43

GetFCDBContent.....	45
GetFoodCount	47
GetFoodCountByProductType.....	49
GetFoodInformation	51
GetFoodList.....	53
GetSupportedTerms	54
Error Messages and Error Codes.....	55
References	59

Introduction

The preliminary specifications for the EuroFIR Web Services were originally created by the European Food Information Network (EuroFIR project, 2005-2010). The EuroFIR project was a Network of Excellence funded by the European Commission's Research Directorate General under the "Food Quality and Safety Priority" of the Sixth Framework Programme for Research and Technological Development.

Objective

This specification defines the EuroFIR Web Services, version 1.2. The objective of the specification is to ensure that food composition data can be interchanged using standardized methods with a network Web Services (i.e. the EuroFIR Web Services) implemented by the EuroFIR partners. This specification defines the rules and requirements for the implementation and the user interface of these Web Services.

The XML schemata are published on the EuroFIR AISBL website (1). The response including FDTP or MDTP is required to be validated against these XML schemata.

The EuroFIR Web Services support Simple Object Access (SOAP) protocol (2).

Important source of information can be found in the EuroFIR Thesauri (3), which is a set of standard vocabularies. Each thesaurus consists of a set of concepts that may be arranged within a hierarchy. A concept is represented by a main descriptor – a term representing the concept – and is generally further described with a scope note, additional information, synonyms and related terms. All thesauri are available on the EuroFIR technical website (1) and updated regularly.

The EuroFIR Web Services support the character encoding UTF-8.

Background

Understanding this specification requires conversance with following documents:

- Proposal for structure and detail of a EuroFIR standard on food composition data: I: Description of the standard.(4)
- Proposal for structure and detail of a EuroFIR standard on food composition data: II: Technical Annex (5); referenced in this document as "Technical Annex".

- EuroFIR Web Services - Food Data Transport Package, Version 1.4. (6); referenced in this document as “FDTP”
- LanguaL Food Description Thesaurus 2012 (or later) (7); referenced in this document as “LanguaL”
- EuroFIR Web Services: Background report (8); referenced in this document as “Background report”

The harmonization and the standardization of the food composition data and the compilation procedures is a long-lasting process: things change and evolve over time and consequently the documents are not perfect or entirely coherent with each other. Thus, interpretation is sometimes needed. As the central point of the EuroFIR Web Services is sending data in the form of the FDTP, enabling this guides the interpretation. Thus, the 'interpretation order' for the implementation of these EuroFIR Web Services is:

- this document
- FDTP
- Technical Annex
- other above mentioned documents

This documentation should be used together with the technical documentation available on the EuroFIR AISBL website (1). This documentation includes following:

- EuroFIR Web Service description in WSDL
- XML schemata for the FDTP
- XML schemata for the MDTP (MDTP is explained later in this document)
- XML schemata for the Food Data Query Language (FDQL is explained later in this document)

Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 (9).

This specification refers to standard vocabularies and versions shown in Table 1. This specification uses also several other naming standards presented in Table 2.

Table 1. Standard vocabularies and versions

Thesaurus	Version	Reference
EuroFIR Acquisition Type Thesaurus	1.1	(3)
EuroFIR Component Thesaurus	1.4	(3)
EuroFIR Value Type Thesaurus	1.1	(3)
EuroFIR Unit Thesaurus	1.1	(3)
EuroFIR Matrix Unit Thesaurus	1.3	(3)
EuroFIR Method Type Thesaurus	1.1	(3)
EuroFIR Method Indicator Thesaurus	1.3	(3)
EuroFIR Reference Type Thesaurus	1.1	(3)
LanguaL Thesaurus	2012	(7)

Table 2. Other naming standards

Naming standard	Date	Reference
ISO 639. Code for the representation of the names of languages	1988	(10), see also Technical Annex
ISO 3166-1 Codes for the representation of names of countries and their subdivisions	1997	(11), see also Technical Annex
RFC 3066: Tags for the Identification of Languages. (XML language tags)	2001	(12)

General Terms

Web services

This specification uses the term 'Web services' referring to the aggregation of different EuroFIR Web Services specified in this document.

Partner Web service

This specification uses the term 'Partner Web service' referring to one Web Service as a part of Web services; This Web service is provided by Web service provider.

Web service providers

This specification uses the term 'Web service providers' referring to the members of the EuroFIR AISBL implementing the Web services.

User application

These Web services are designed to be used by different search tools for searching and retrieving the food composition information and processing it. They are all referred to as 'user application'.

User application providers

This specification uses the term 'User application providers' referring to all different parties implementing and managing the user applications.

End-user

This specification uses the term 'end-user' referring to an abstraction of the group of persons using the food composition information (via user applications).

FCDB

Food Composition Database (FCDB) is a database with information about foods and their composition. In this specification, the term refers to FCDBs maintained by the Web service providers.

FDTP

Food Data Transport Package used in the data interchange (6)

MDTP

Metadata Transport Package is a data transportation package used for providing information about the Partner Web service and its FCDB. (See chapter Metadata Transport Package (MDTP) in this document)

For other terms and acronyms see the Background report. If some standard has several versions and it is essential to know which version we are referring to, the reference is given in this specification when the standard is mentioned (e.g. SOAP “envelope” (2))

Process Flow

1. Request reception and delegation
2. Authentication
3. Query validation
4. Query interpretation and translation to FCDB query/queries
5. FCDB query/queries
6. FDTP compilation
7. Response compilation and sending

This specification defines processes 1 - 3 and 7. It also includes semantic rules for the tasks in the process 4 but detailed implementation is partner specific. The processes 5 and 6 are entirely partner specific. All processes require appropriate error handling: See Error messages and error codes.

Web Service Request

Web service requests SHALL use SOAP "envelopes" (2).

The request

- MUST use either the UTF-8 (13) encoding.
- MUST be a well-formed (XML 1.0) document (14)
- MUST contain method element for Web service identification
- MUST contain parameters for authentication.

The request in the envelope is posted to a URL.

```
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope"
xmlns:eur="http://eurofir.webservice.namespace">
  <env:Body>
    <eur:methodName>
      <eur:parametername>value</eur:parametername>
    </eur:methodName>
  </env:Body>
</env:Envelope>
```

The first element inside the body MUST be the methodName element. (i.e. the name of the Web Service, e.g.eur:GetFoodInformation) Each request parameter SHALL be a single child of that (like e.g. elements eur:api_userid, eur:api_permission...). The method element identifies the Web Service where the request is delegated to.

Web Service Response

Web service responses SHALL use SOAP "envelopes" (2)

The response

- MUST use either the UTF-8 (13) encoding.
- UTF-8 is RECOMMENDED as FDTP uses that as default encoding
- MUST be a well-formed (XML 1.0) document (14)

A return to the response will be either:

1. Normal response with a FDTP
2. Normal response with a Metadata Transfer Package (MDTP)
3. Error message

Normal Response

FDTP or MDTP included in the normal response MUST be valid i.e. validated against the appropriate XML schemata.

FDTP

```
<env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">
  <env:Body>
    <EuroFIRServiceFDTPResponse xmlns="http://eurofir.webservice.namespace">
      <EuroFIRFoodDataTransportPackage>
        <!--put your data here-->
      </EuroFIRFoodDataTransportPackage>
    </EuroFIRServiceFDTPResponse>
  </env:Body>
</env:Envelope>
```

Metadata Transfer Package (MDTP)

```
<env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">
  <env:Body>
    <EuroFIRServiceMDTPResponse xmlns="http://eurofir.webservice.namespace">
      <EuroFIRMetaDataTableTransportPackage>
        <!--put your data here-->
      </EuroFIRMetaDataTableTransportPackage>
    </EuroFIRServiceMDTPResponse>
  </env:Body>
</env:Envelope>
```

Error Message

```
<env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">
  <env:Body>
    <env:Fault xmlns:flt="http://www.w3.org/2003/05/soap-envelope">
      <faultcode>env:SOAP standard fault code (like e.g. env:Server)</faultcode>
      <faultstring>Some Web service error message (like e.g. Invalid message signature
checking)</faultstring>
      <detail>
        <EuroFIRServiceFault:EuroFIRServiceFault
xmlns:EuroFIRServiceFault="http://eurofir.webservice.namespace"
xmlns="http://eurofir.webservice.namespace">
          <errorcode>Web service errorcode (like e.g. E2022)</errorcode>
          <reason>Some detailed error description</reason>
        </EuroFIRServiceFault:EuroFIRServiceFault>
      </detail>
    </env:Fault>
  </env:Body>
</env:Envelope>
```

faultcode

SOAP standard fault codes.

faultstring

Web Services Error message (see Error messages and error codes)

detail

Additional information about the error. For example the Web Services error code and a description of the reason for the error (see Error messages and error codes).

Advice for the developers

```
<eur:fdql_sentence><![CDATA[<?xml version="1.0" encoding="UTF-8"?>
<FDQL_Sentence>
  <MetaData>
    <SchemaVersion>1.0</SchemaVersion>
    <Schema>EuroFIR_Web_Service_FDQL_Sentence_version_1_2.xsd</Schema>
  </MetaData>
  <SelectClause>
    <FieldName>Content</FieldName>
  </SelectClause>
</FDQL_Sentence>]]></eur:fdql_sentence>
```

Ensure that your implementation really encodes the FDQL with UTF-8 as other encodings will cause complications in the user applications. It is usually not enough to only add the correct UTF-8 header. Moreover, the default settings of the implementation tools (e.g. Java) seem to produce encoding which is not UTF-8 and it has to be defined in the programming code that UTF-8 is used in every phase when the XML is produced. However, when the data contains only simple ASCII characters, it is very difficult to see whether the encoding is really UTF-8. Use UTF-8 compatible editor for creating test cases and it is RECOMMENDED to use non-ASCII characters in the testing.

The Food Data Query Language (FDQL) statement is often part of the request. In these cases, it is RECOMMENDED to use CDATA sections for the query sentence.

Web Services Authentication

The main principle of the Web services authentication is that Web service providers control the access to their resources. All requests MUST be authenticated. The authentication is done at the user application level, not at the end user level. However, the user applications SHALL have their own authentication for the end users but that is out of the scope of this specification.

The Web service provider delivers the needed access keys to the user application provider.

Authentication keys

The EuroFIR Web service provider SHALL deliver three keys to the user application provider:

1. User application identification key
2. User application secret key
3. User application permission key is currently not used for setting permissions but for support for multiple databases

User application identification key

This key MUST be used for the user application identification and it MUST be included with each request.

- A 20-character, alphanumeric sequence (for example 1079812AWLTL45NLS3IP).
- Parameter name: api_userid

User application secret key

This key MUST be used for the calculation of the signature.

- A 40-character sequence (for example 1+09ku7jht/akHT2LO86zxcMK912hT/hIKU46QWC).
- Referred in the procedures as 'SECRET'.

User application permission key

When implemented this key MUST be used for defining different access profiles and it MUST be included with each request.

- Currently, this is used for defining from which country database the data is being requested (e.g., si [for Slovenia])
- Parameter name: api_permission

Signing the request

Each request MUST be signed using a hash-based message authentication code. This is done by calculating a hashed parameter using the user application secret key and the parameters in the request.

- Hashing uses the SHA-1 Message-Digest Algorithm (SHA1) (15)
- Parameter name: api_signature

Signature calculation procedure

- Sort the argument list into alphabetical order based on the parameter name; e.g. foo=1, bar=2, baz=3 sorts to bar=2, baz=3, foo=1
- concatenate the shared secret and argument name-value pairs; e.g. SECRETbar2baz3foo1
- calculate the SHA1() hash of this string
- append this value to the argument list with the name api_signature, in hexadecimal string form; e.g.
api_signature=1f3870be274f6c49b3e31a0c6728957f

Checking the request

When a request is received by the Web service, the authentication is checked. If the request does not pass the checking, access is revoked and the proper error message is returned (see Error messages and error codes).

Risk management of the identification keys

The user application provider MUST control the access to these keys and they SHALL NOT be provided to end users. These keys are used inside the Web services and the user applications and they MUST be stored so, that only those applications can access them. Encryption is RECOMMENDED in the storing of the keys.

Authentication errors

See Error messages and error codes

Advice for the developers

Ensure that your implementation uses the UTF-8 encoding. The different encodings produce different api_signatures. As the user application is expecting to receive UTF-8, it calculates the signature with UTF-8. However, if the original request was signed with a different encoding, the signatures will not match and the request will fail. See the advice in the previous chapter.

Food Data Query Language (FDQL)

Web services use a Food Data Query Language (FDQL) for the defining of the food information content and the search options used in the requests. This language resembles SQL but is more abstractive in its nature because it is not connected to any existing data model in any specific FCDB. Thus, the FDQL is connected with an abstract data model and the FDQL sentence needs to be translated to an actual query (or queries) which may differ in each individual implementation. This specification does not define what the actual query should be – it only defines the rules for the translation. Some of the rules are defined by the FDQL features and they are supplemented by the semantic rules.

FDQL Data Model

This data model defines the entities and their relations in the way that a FDQL sentence can be translated to FCDB queries. The FDTP uses the same entities, but their relations are structured slightly differently. However, the current data model can be used for constructing the FDTP. We know also that different FCDBs may be designed differently: it is entirely possible that in some FCDB e.g. food names are in a separate table and in some other FCDB they may be columns of the food table. Still, in both cases the FDQL sentence can be translated into FCDB query (or queries) and produce the standardized FDTP. These are the only criteria for the interface.

The entities are connected with relations, which use identifiers of some kind (usually a pair of primary key-foreign key). However, the only public identifiers that are visible outside in a sense that they may be used in the FDQL queries are:

- the original food id (origfdcd in the FDTP)
- the original component code (origcpcd in the FDTP)

This means that even there are other private identifiers connecting the entities (like e.g. food with foodnames or component value with method), these identifiers are not available in the FDQL sentences - we just assume that the

entities are connected by some mean. Consequently, all associations are predefined and fixed: the entities cannot be linked differently in different FDQL sentences (see Semantic rules for FDQL translation) - this is also the biggest difference between a subset of SQL and FDQL.

The main entities are food, component and component values (Figure 2).

These main entities form groups related with them entities:

- Food related entities (Figure 3)
- Component related entities (Figure 4)
- Component Value related entities (Figure 5)

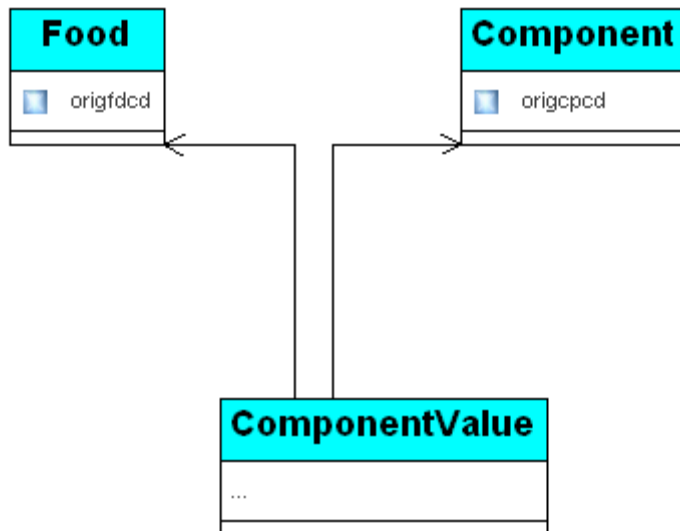


Figure 2. Main entities.

This basic structure forms the basic linking for the FDQL: Component Value is always linked with some Food and some Component. All other entities are subordinate to these main entities: they are additional information which may exist or may not exist (some fields are mandatory some are optional). This is also an essential feature of the semantic rules in the FDTP, these same main entities – now XML elements – form the main structure of the package: Food → Component → Component Value.

There are also some entities, which are not related with the main entities, referred here as assisting entities (Figure 6).

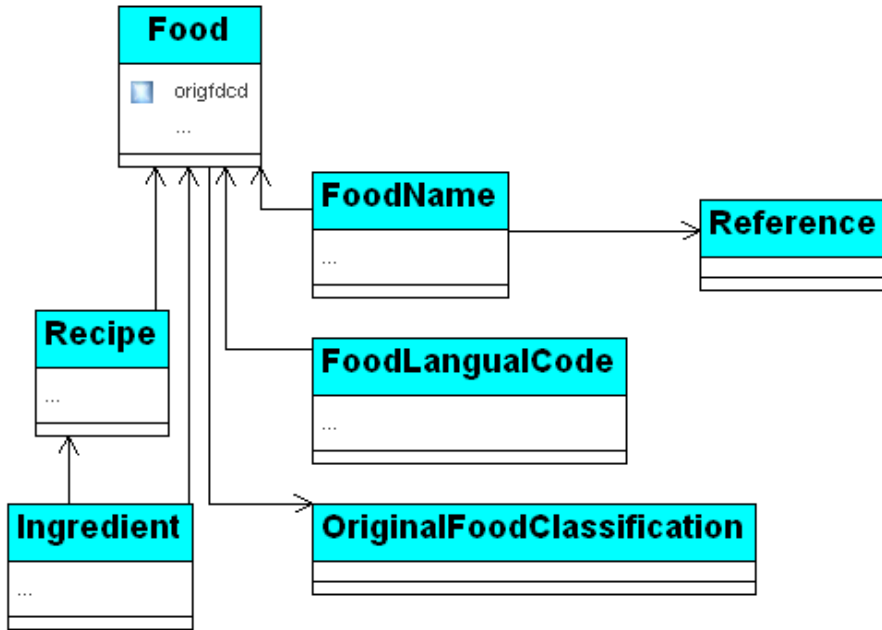


Figure 3. Food related entities.

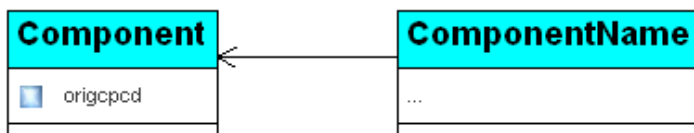


Figure 4. Component related entities.

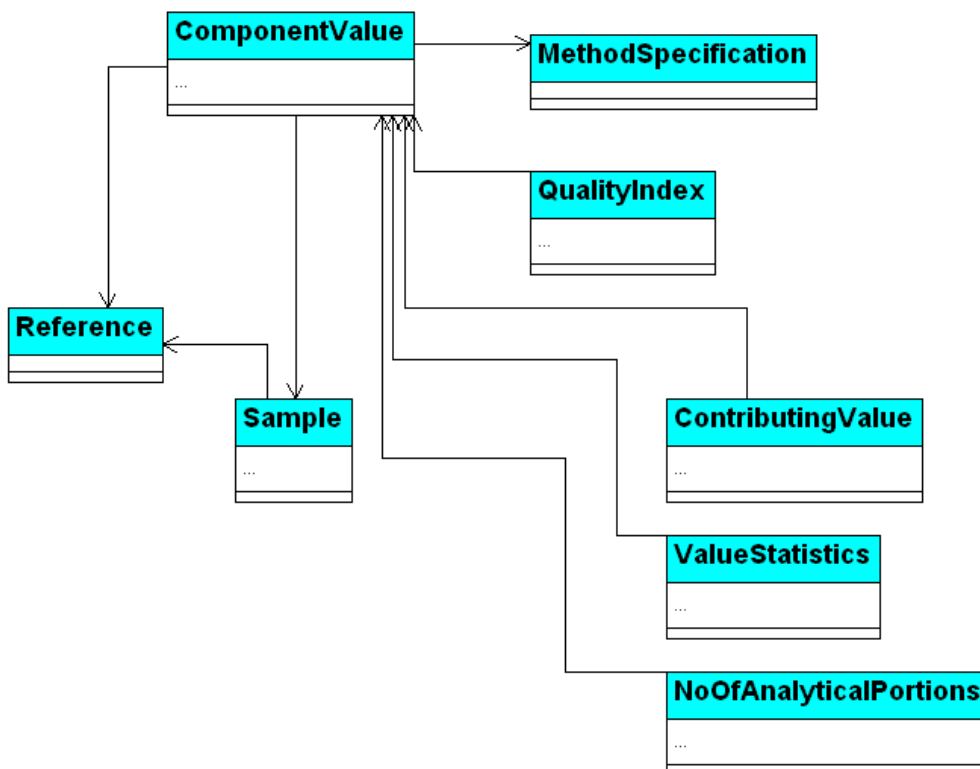


Figure 5. Component Value related entities.

Content

RetentionFactor
...

Figure 5. Assisting entities.

We use a term 'field' referring to an element which is usually a column in a relational data base table. In the FDTP, which uses XML, these 'fields' are sometimes presented as 'elements' and sometimes 'attributes' and it would be confusing to use the term 'attribute' instead of 'field'. Note that this specification does not contain all existing fields or their specification and FDTP and Technical Annex should be used as the comprehensive reference.

FDQL Data Model Restrictions

Standard vocabularies are not included

The FDTP uses several standard vocabularies but they are not included in the FDTP - they are only referred to. This means, for example, that the FDTP includes only the code of the method type 'D' but not the whole term 'Aggregation of contributing analytical results'. The Web services follow the same convention and all terms from the standard vocabularies have been excluded. This also means that these fields can not be used in the FDQL sentences even some of them were used in the use cases. Only terms that have been included are the original food classification (origgpcd in the FDTP) and the original component name (origcpnm in the FDTP) which are local and not part of any the standard vocabulary. However, the user applications may of course utilize the terms from the standard vocabularies as a part of their query interface or presenting the query results in the more understandable way.

FoodLanguaLs cover food description

Food LanguaL descriptors cover all harmonized food classifications and general food description (in the Technical Annex). The Technical Annex defines several property identifiers (like e.g.'COOKMETH') for food classification and food description but in the FCDBs they may be organized in several tables. However, as the facet of all LanguaL codes can be detected by the first letter (e.g. 'C1234' belongs to LanguaL facet C 'Physical State Shape or Form'), they are treated under the same concept (as FoodLanguaLs) and the codes must be mapped in the translation process according to the FCDB structure. Also the FDTP groups them under the

FoodClasses element. Only non-LanguaL classification is the original local food classification.

Limitation of use in different clauses of the FDQL

Current specification includes only Web services with priorities 1-2. This means that some features are not supported. See the FDQL reserved words.

Additional information in the metadata

It was found useful to include optional open parameters with the FDQL sentence. These Entity-Attribute-Values may be used by the Web service providers and user applications for features not included in these specifications. A new element AdditionalInformation was added to the metadata-section.

Component Values allowed in WHERE

ComponentValues are allowed in the WHERE –clause.

Additional information in the original food classification

FDQL does not define any specific food classification. Some FCDBs may use several different food classifications. An attribute has been added to the element ClassificationConditionField. The attribute targetClassification can be used for specifying the classification. However, this specification does name any specific classifications which should be supported or reserved words for any specific classification.

Experimental feature using the energy distribution

Energy distribution gives summary information of a food item such as how many percentages of the total energy are from the proteins. Typically, this information is given for proteins, carbohydrates and fats and is an important part of the nutrient profile. However, the current rules for the calculation of the energy are not coherent and usually FCDBs do not contain values except total energy. Thus, the feature is only experimental.

FDQL Reserved Words

The FDQL reserved words are either words defining the FDQL language structure or the data content. The FDQL language structure is defined by the XML schema with the element names and the attribute names , which are all

reserved words (see FDQL Implementation). The language content is based on the concepts used in the FDTP and in the Technical Annex. There are also minor differences between FDTP and the Technical Annex - in such cases we use the convention of the FDTP. Note that the FDTP and Technical Annex include several fields which may be included in the FDTP content but are not included in the FDQL (like e.g. remark). The reserved terms are also included in the XML schema (see FDQL implementation).

Table 3. Food related terms. =allowed, =not allowed























Entity	Term	SELECT -clause	WHERE -clause	ORDER BY- clause	Comment
Food	origfdcd				CommonConditionField
FoodLangual Code	FoodIdentifier Langual				WHERE-clause defines the search scope (BT/NT), ClassificationConditionField
FoodName	FoodName				NameConditionField. The language is defined by the language attribute
OriginalFood Classification	origgpcc				ClassificationConditionField
Recipe	Recipe				
Ingredient	Ingredient				
Food	FoodAll				Group term. See Semantic rules
Food	FoodAll Mandatory				Group term. See Semantic rules
Food	FoodAllMinimum				Group term. See Semantic rules
Food	FoodList				Group term. See Semantic rules

Table 4. Component related terms. =allowed, =not allowed

Entity	Term	SELECT-clause	WHERE-clause	ORDER BY-clause	Comment
Component	ecompid				WHERE-clause defines the search scope (BT/NT), ClassificationConditionField
Component	origcpd				CommonConditionField
ComponentName	origcpnm				NameConditionField. The language is defined by the language attribute
Component	ComponentAll				Group term. See Semantic rules
Component	ComponentAll Mandatory				Group term. See Semantic rules
Component	ComponentAll Minimum				Group term. See Semantic rules
Component	ComponentList				Group term. See Semantic rules

Table 5. Component Value related terms. =allowed, =not allowed



Entity	Term	SELECT-clause	WHERE-clause	ORDER BY-clause	Comment
ComponentValue	SelectedValue				ValueConditionField
ComponentValue	Minimum				ValueConditionField
ComponentValue	Maximum				ValueConditionField
ComponentValue	Mean				ValueConditionField
ComponentValue	ComponentValue All				Group term. See Semantic rules
ComponentValue	ComponentValue AllMandatory				Group term. See Semantic rules
ComponentValue	ComponentValue AllMinimum				Group term. See Semantic rules
QualityIndex	QualityIndex				Group term for all quality indices. See Semantic rules
MethodSpecification	Method Specification				Group term for all MethodSpecification fields See Semantic rules
Sample	Sample				Group term for all sample fields. See Semantic rules
ContributingValue	Contributing Value				Group term for all contributing values. See Semantic rules
ValueStatistics	ValueStatistics				Group term for all value statistics. See Semantic rules
NoOfAnalytical Portions	NoOfAnalytical Portions				Group term for Number of analytical portions. See Semantic rules
Reference	ValueReference				Group term for value reference. See Semantic rules
Reference	Method Reference				Group term for method reference. See Semantic rules
EnergyDistribution	Energy distribution				Experimental ValueConditionField. See semantic rules.

Table 6. Metainformation terms. =allowed, =not allowed

Entity	Term	SELECT- clause	WHERE- clause	ORDER BY- clause	Comment
Metadata	Content				Group term. See Semantic rules
Metadata	Count				Group term. See Semantic rules
Metadata	AvailableFoods				Group term. See Semantic rules
Metadata	Available Components				Group term. See Semantic rules
Metadata	SupportedSelect Terms				Group term. See Semantic rules
Metadata	SupportedWhere Terms				Group term. See Semantic rules
Metadata	SupportedOrder ByTerms				Group term. See Semantic rules

FDQL Implementation

The FDQL sentence in the request is formed in XML and it MUST be validated by EuroFIR Web Service FDQL Sentence Schema. This XML schema provides aids for checking the FDQL sentence syntax before translating it to a partner specific FDBC query/queries.

- MUST use either the UTF-8 (13) encoding.
- UTF-8 is RECOMMENDED as FDTP uses that as default encoding in the future.
- MUST be a well-formed (XML 1.0) document (14)

FDQL Sentence Structure in XML

The XML schema (published in the EuroFIR AISBL website (1)) gives the detailed documentation of the FDQL sentence. The FDQL sentence has four main elements:

1. MetaData
2. SelectClause
3. WhereClause
4. OrderByClause

MetaData

MetaData describes the XML schema and its version. This XML schema can be used in the validation.

SelectClause

SelectClause defines the needed information content.

WhereClause

WhereClause may restrict the retrieved information content.

OrderByClause

WhereClause may order the retrieved information content.

Example of the FDQL Sentence

NOTE: The example is just an arbitrary example and it is constructed for demonstration of the language features. Some features like e.g. giving the search condition with component value range do not belong to the current implementation of the Web services even they are features of the FDQL. The XML schema for the FDQL sentence and the XML schema used in the reserved words are published in separate documents in the EuroFIR AISBL website (1).

```

<FDQL_Sentence>
  <MetaData>
    <SchemaVersion>1.2</SchemaVersion>
    <Schema> EuroFIR_Web_Service_FDQL_Sentence_version_1_2.xsd</Schema>
  </MetaData>
  <SelectClause>
    <FieldName>origfdcd</FieldName>
    <FieldName>FoodName</FieldName>
  </SelectClause>
  <WhereClause>
    <Condition xsi:type="T_CommonCondition" logicalOperator="AND">
      <CommonConditionField>
        <FieldName>origfdcd</FieldName>
      </CommonConditionField>
      <ConditionOperator>=</ConditionOperator>
      <ConditionValue>1234</ConditionValue>
    </Condition>
    <Condition xsi:type="T_CommonCondition" logicalOperator="AND">
      <NameConditionField language="en">
        <FieldName>FoodName</FieldName>
      </NameConditionField>
      <ConditionOperator>LIKE</ConditionOperator>
      <ConditionValue>Somethi%</ConditionValue>
    </Condition>
    <Condition xsi:type="T_InCondition" logicalOperator="OR">
      <ClassificationConditionField searchScope="BT">
        <FieldName>FoodIdentifierLanguag</FieldName>
      </ClassificationConditionField>
      <ConditionOperator>IN</ConditionOperator>
      <ConditionValue>A1234</ConditionValue>
      <ConditionValue>A3456</ConditionValue>
      <ConditionValue>G1234</ConditionValue>
    </Condition>
    <Condition xsi:type="T_CommonCondition" logicalOperator="AND">
      <NameConditionField language="en">
        <FieldName>FoodName</FieldName>
      </NameConditionField>
      <ConditionOperator>LIKE</ConditionOperator>
      <ConditionValue>Porridge</ConditionValue>
    </Condition>
    <Condition xsi:type="T_BetweenCondition" logicalOperator="AND">
      <ValueConditionField ecompid="VITA" matrixUnit="D" unit="ug">
        <FieldName>SelectedValue</FieldName>
      </ValueConditionField>
      <ConditionOperator>BETWEEN</ConditionOperator>
      <ConditionValue>1</ConditionValue>
      <ConditionValue>100</ConditionValue>
    </Condition>
  </WhereClause>
  <OrderByClause>
    <OrderByField orderingDirection="ASC">
      <FieldName>origfdcd</FieldName>
    </OrderByField>
    <OrderByField orderingDirection="DESC">
      <FieldName>ComponentName</FieldName>
    </OrderByField>
  </OrderByClause>
</FDQL_Sentence>

```

Semantic Rules for the FDQL Sentence Translation

General Interpretation Rules

Main entities and their subordinate entities

The FDQL Data model defines the main entities: There are three main entities

1. Food
2. Component
3. Component Value

As shown in the data model, other entities, the subordinate entities, are associated with these main entities. In the FDTP we again find these same main entities. They are, however, presented with XML elements, and they form a Food-oriented structure: Food → Component → Component Value.

This Food-oriented structure guides the FDQL sentence translating process: first we try to find out which Foods are needed in the process (if any), then we see which Components are needed (if any) and finally which Component Values should be retrieved.

Inclusion principle: No subordinate entity without the main entity

A subordinate entity may not exist alone: e.g. if there is a Food name, there must also be Food.

This gives us the inclusion principle: if the subordinate field is used in the FDQL sentence, the translation MUST include the related main entity - even the main entity is not mentioned in the FDQL sentence.

Return only what is requested

If some entity (main or subordinate) is not requested it, SHOULD NOT be used in the translation or in the data retrieval. For example, if ComponentValueAllMinimum is requested, do not include the Sample.

Supplement principle: Group term takes everything available from the entity

There are several group terms (see Reserved words) which imply retrieving the whole entity content. Using the group term implies the supplement principle: If the group term relates to a main entity, then take this main entity and all its subordinate entities - take every available field from them

(mentioned or not). If the group term relates to a subordinate entity, then take every available field from this entity. Following this supplement principle is REQUIRED.

For example the group term 'AllComponentValue' means that every entity from Method specification to Reference has to be included. Equally, if we use the group term 'QualityIndex', this implies taking all different fields of the QualityIndex. Note, that sometimes other rules limit effect of the supplementation principle. For example, FoodAllMinimum defines that not all fields or entities are included even the supplementation principle says that we should take everything.

Respect the FCDB rules

When the food information is retrieved, it will be eventually compiled in the form of the FDTP. Using the FDTP sets certain minimum requirements, meaning that some mandatory entities have to be included even not mentioned in the FDQL sentence. Currently there are minimum requirements and then a somewhat larger set of fields and entities defined in the FDTP and in the Technical Annex as mandatory. Following the FDTP requirements is REQUIRED. The only exceptions are specifically mentioned (e.g. like MetaInformation and Component list) in the semantic rules of the specific Web services. Moreover, the FDTP structure also defines the content and the order of the elements and SHALL NOT be overruled by the order of the fields in the SELECT-clause.

Joins Between the Entities

The existing food composition information is quite heterogenic: Many of the fields are optional and even mandatory information might not be comprehensive in the time being. FCDBs and their management systems are in many cases in the middle of the development process and the harmonization of the food information is not yet complete. This means that different Web service providers may support different selections of fields and even the field is supported the information might not be comprehensive, for example, for every food. To summarize: we do not have all information about all fields available.

In general, we try to provide as much information that is possible. This means that there will always be "holes" in the datasets - or NULLs if we use relational database terminology. If we think our main entities, the only thing which always exists: the Component Value has always the Component and the Food. Still, there may be Foods without Component Values. These facts give us five entity joining principles:

1. The left outer join is the basic join between the main entities Food and Component Value (from Food to Component Value). Food may or may not have Component Values but a Component Value cannot exist without Food.

2. If there are, however, WHERE-clause conditions limiting Component Values, the join between the Food and Component Value is inner join.
3. The left outer join is basic join between the main entities Component and Component Value (from Component to Component Value). Component may or may not have Component Values but a Component Value cannot exist without Component. The FDTP is, however, Food-oriented meaning that Components without any Component Values will never be retrieved –Component List and Metainformation are the only exceptions. Thus, normally the inner join can be used between the Component and the Component Value.
4. The left outer join is basic join between the main entities and its subordinate entities (from the main entity to the subordinate entity).
5. If there are, however, WHERE-clause conditions limiting the subordinate entity, the join between the main entity and the subordinate entity is inner join.

The principles 1-3 mean that if we have, for example, AllFromFood, AllFromComponent, AllFromComponentValue, we shall take all foods with or without Component Values (i.e. even the Component Values were missing) (principle 1, left outer join). However, the FDTP is food oriented and so we will not be able to take Components without any Foods (principle 2) In comparison, if there are WHERE-clause conditions e.g. for the Component Value, the Foods without any Component Values would automatically be excluded from the dataset (principle 2, inner join). However, if the WHERE-clause was only for the Food, then that would not have any effect on the Component Value: so again we take Foods even they had no Component Values (principle 1 left outer join) – that Component Value element would be empty in such case.

Another example: We have AllFromFood, meaning that all Food names will be retrieved (principle 4 left outer join): those with the scientific name or without the scientific name. However, if we had a WHERE-clause condition concerning the scientific name, we would take only Foods having that scientific name (excluding Foods without any scientific name or with scientific names not matching the condition) (principle 5, inner join).

Following these joining principles is REQUIRED. However, as the FCDBs are different (with different table structures, different database management systems etc.), there may be situations where the principles should not be taken literally. Sometimes, for example, the outer joins could be implemented by looping and inner joins – only the result has to be the same. The key issue is to respect the purpose of the principles and this can be achieved with various ways.

It should be stated that of course the join should not be made if the entity is not needed at all: if the FDQL sentence uses only foods, there is no need to make the unnecessary joins with the Component (or Component Values).

NULLs will produce empty elements (or attributes)

It is obvious that the left outer joins produce many NULL values. For example, if we join a Component Value with a Sample and there is no Sample, all Sample fields will be NULL. These NULLs are translated to empty elements (or attributes) when the FDTP is compiled i.e NULLs are treated as empty. However, in many cases including the empty elements to the FDTP is not necessary (or not allowed) – especially if they are not required elements or attributes.

Validation of the FDQL Sentence

The XML schemata define the FDQL sentence structure (published in the EuroFIR AISBL website (1)). Moreover, they define, which of these terms could, in general, be used in SELECT, WHERE and ORDER BY-clauses (see also FDQL reserved words). Thus, the XML schema is used for validation. If the term used in the FDQL sentence is not among the reserved terms or if the FDQL sentence is not proper, the Web Service SHALL interrupt and send a proper error message (See Error messages and error codes). In addition, different Web Services may have additional restrictions: some terms are not allowed in every Web service.

The validating abilities of the XML schemata are not comprehensive. This means that some term may pass the validation but is not supported in the FCDB (e.g. like Original Food Classification). Moreover, the schema is able to validate whether the SELECT-clause is empty but not whether the fields are suitable for the task. For this, there are semantic rules and some Web services may have their own, additional, semantic rules. In addition, the semantic rules may cause complications needing further checking. The rules might e.g. omit all the fields in the SELECT-clause (as not supported fields). To prevent this, there is yet another rule: SELECT-clause SHALL NOT be empty after potential omitting. In such case, the Web service is interrupted and an error message is sent (See Error messages and error codes). Moreover, the current XML schemata do not cover all standard vocabularies and it is not possible to validate e.g. whether a certain Language code exists or not. Checking these is left outside the scope of the current validation procedures – but of course no data will be retrieved with a non-existing code.

Omitting fields is possible in the SELECT-clause and in the GROUP BY -clause

There may be occasions where some field is not supported by some Partner Web Service for the reasons explained before. This may happen even the field is one of the reserved words i.e. it passes the XML schema validation. If the field is not supported, it SHALL be omitted from the SELECT and GROUP

BY -clauses but the retrieval process SHALL continue normally. Leaving out unsupported fields does not have any effect.

Omitting fields is not possible in the WHERE-clause

Similarly, some WHERE-clause fields may not be supported. Contrasting with the rather harmless effect on the SELECT-clause, omitting a field from WHERE-clause would change the result considerably. Thus, if the field is not supported, the Web service will be interrupted and return a proper error message (See Error messages and error codes).

WHERE-clause Evaluation

As stated before, the entities have predefined associations which will be commonly implemented by SQL joins or SQL WHERE clauses in the translation process. Contrasting with these SQL WHERE clauses or joins, the FDQL sentence WHERE-clause SHALL be interpreted always restrictive: i.e. FDQL sentence WHERE-clause can not expand the original joins. This is equal with the situation where the whole FDQL sentence WHERE -clause is connected with the AND-operator with the previous predefined SQL joins. This means that the first logicalOperator of the Condition-elements will always be interpreted as 'AND', because it links all Condition-elements with the joins. (We can not leave it empty because the XML schema says that the logicalOperator attribute is mandatory for all Condition-elements).

The FDQL sentence WHERE-clause may contain several Condition-elements. Condition-elements have a REQUIRED logicalOperator (AND|OR|AND NOT|OR NOT) defining the connection between two Conditions. All Condition-elements are evaluated from top to bottom. There is no other precedence between logicalOperators other than the order of the Condition-elements. This means that the order in which the Condition-elements are presented has in many cases a significant effect on the query result and this has to be taken into account in the query design. All other logicalOperator-attributes connect two consecutive Condition-elements. Following this order is REQUIRED.

Example of the evaluating order of Condition-elements (simplified elements)

```
<Condition logicalOperator="AND">foo</Condition>  
<Condition logicalOperator="OR">bar</Condition>  
<Condition logicalOperator="AND NOT">something</Condition>  
<Condition logicalOperator="OR NOT">something2</Condition>
```

is interpreted like

```
(join-operations) AND (((foo OR bar) AND NOT something) OR NOT something2)
```

Ordering with the ORDER BY -clause

As stated many times, the FDTP is Food –oriented. This means that we are not able to order the data content any way we would like to. However, we may use ORDER BY for subordinate entities inside Food, inside Component and inside Component Value. There are also some (MetaInformation) services which are not so tightly bound with the FDTP structure.

Food related semantic rules

FoodAll

Term "FoodAll" is a group term, which refers to all existing food related fields in the FCDB in the Technical Annex. There may be, however, some limitations in the implementation because the rules for the implementation may be incomplete all of them in the FDTP. The term is allowed only in the SELECT-clause.

FoodAllMandatory

Term "FoodAllMandatory" is a group term, which refers to all food related fields in the FCDB defined in the Technical Annex defined as mandatory. There may be, however, some limitations in the implementation because the rules for the implementation may be incomplete all of them in the FDTP (e.g. reference). The term is allowed only in the SELECT-clause.

FoodAllMinimum

FoodAllMinimum is a group term, which refers to all food related fields in the FDTP defined in the minimum requirements. Allowed only in the SELECT-clause.

FoodIdentifierLanguaL

Term "FoodIdentifierLanguaL" refers to all LanguaLs in the FoodClasses (see also FoodLanguaLs in the FDQL Data model). Used allways with a searchScope attribute: Broader term (BT) / Narrower term (NT). As explained in the Background report, the narrower term refers to a search by a specific LanguaL code and the broader term is hierarchical. A broader term always yields results to all narrower terms as well.

EuroFIR Food Classification

EuroFIR Food Classification is part of the LanguaL A facet (Product type, sub-tree under code A0777) and is treated as part of the LanguaLs (see FoodIdentifierLanguaL).

OriginalFoodClassification

Term "origppcd" refers to the original food classification scheme. Used always with a searchScope attribute: Broader term (BT) / Narrower term (NT). The implementation of the original food classification varies because as an optional classification it may not exist at all. Even if the classification exists the classification schemes are different. Thus, the implementations differ and especially whether the attributes NT (narrower term) or BT (broader term) have any effect depends very much about the classification.

FoodNames

FDQL sentence uses FoodNames with a mandatory language attribute. Implementing this is REQUIRED. The FDQL sentence uses a language attribute (comparable with xml:lang see RFC 3066 (12)) which is equal with the FDTP convention (ISO 639 2 character code for language (10) plus an optional 2 character standard ISO country code (10)). Language "tx" is reserved for scientific names. The existence of the language attribute is validated with the schema but the content is not. The translation is RECOMMENDED to check that the sentence language code is one of those available (i.e. supported) in the FCDB (See Error messages and error codes).

FoodNames may be preferred or synonyms (see the FDTP). However, these are not separated in the FDQL sentence. Only the preferred name is REQUIRED in the implementation of the query translation. There is no term or attribute available in the FDQL sentence specifying whether some name e.g. in the WHERE-clause is preferred or synonym.

Recipe

Term "Recipe" refers to the whole recipe content of the FDTP and it is a group term meaning all elements under the Recipe element. Term is allowed only in the SELECT-clause.

FoodList

Term "FoodList" is a group term, which refers to the FoodDescription element (without Recipe) from FDTP. The term is used only in the Web Service GetFoodList.

Component Related Rules

ComponentAll

Term "ComponentAll" is a group term, which refers to all existing component related fields in the FCDB defined in the Technical Annex. There may be, however, some limitations in the implementation because the rules for the

implementation may be incomplete all of them in the FDTP. The term is allowed only in the SELECT-clause.

ComponentAllMandatory

Term "ComponentAllMandatory" is a group term, which refers to all component related fields in the FCDB defined in the Technical Annex defined as mandatory. The term is allowed only in the SELECT-clause.

ComponentAllMinimum

ComponentAllMinimum is a group term, which refers to all component related fields in the FDTP defined in the minimum requirements. The term is allowed only in the SELECT-clause.

OriginalComponentName

The terms "OriginalComponentName" and "origcpnm" refer to the OriginalComponentName. Components do not have any other "official" name than the name (or Term) from the EuroFIR Component Thesaurus. However, as part of all the Standard Vocabularies, this has been excluded from the Web Service data model (see FDQL Data model) and this term are currently not used (use leads to interruption and error message). However, the term is still part of the reserved words in the XML schema and the schema validation does not catch this term.

origcpcd

The terms "origcpcd" refers to the OriginalComponentCode. This field is mandatory in the Technical Annex and FDTP but, however, it may not exist in the FCDB (i.e. the FCDB may be based on entirely on the ecompid - in such case the origcpcd SHALL be interpreted as ecompid).

ecompid

Term "ecompid" refers to the EuroFIR Component Thesaurus. It is used allways with a searchScope attribute: Broader term (BT) / Narrower term (NT). The EuroFIR Component Thesaurus version 1.0 had only one level but since version 1.1 this thesaurus is hierarchical. This means that currently there are hierarchical component group codes used together with the component codes used in the ComponentValues. Thus, the BT/NT should be implemented using similar search logic as with LanguaL codes (FoodIdentifierLanguaL). As explained in the Background report, the narrower term refers to a search by a specific code and the broader term is hierarchical. A broader term always yields results to all narrower terms as well.

ComponentList

Term "ComponentList" is a group term, which refers to ComponentIdentifiers element of the FDTP. The term is used only in the Web Service Get Component List. See Get Component List.

Component Value Related Rules

ComponentValueAll

Term "ComponentValueAll" is a group term, which refers to all existing Component Value related fields (and entities) in the FCDB defined in the Technical Annex. There may be, however, some limitations in the implementation because the rules for the implementation may be incomplete all of them in the FDTP Allowed only in the SELECT-clause.

ComponentValueAllMandatory

Term "ComponentValueAllMandatory" is a group term, which refers to all Component Value related fields (and entities) in the FCDB defined in the Technical Annex defined as mandatory. The term is allowed only in the SELECT-clause.

ComponentValueAllMinumum

ComponentValueAllMinumum is a group term, which refers to all component value related fields (and entities) in the FDTP defined in the minimum requirements. The term is allowed only in the SELECT-clause.

QualityIndex

QualityIndex is a group term, which refers to all QualityIndex related fields. The term is allowed only in the SELECT-clause.

MethodSpecification

MethodSpecification is a group term, which refers to all MethodSpecification related fields. The term is allowed only in the SELECT-clause.

QualityIndex

QualityIndex is a group term, which refers to all QualityIndex related fields. The term is allowed only in the SELECT-clause.

Sample

Sample is a group term, which refers to all Sample related fields. The term is allowed only in the SELECT-clause.

ContributingValue

ContributingValue is a group term, which refers to all ContributingValue related fields. The term is allowed only in the SELECT-clause.

ValueStatistics

ValueStatistics is a group term, which refers to all ValueStatistics related fields. The term is allowed only in the SELECT-clause.

ValueReference

ValueReference is a group term, which refers to all ValueReference related fields. The term is allowed only in the SELECT-clause.

MethodReference

MethodReference is a group term, which refers to all MethodReference related fields. The term is allowed only in the SELECT-clause.

NoOfAnalyticalPortionsValue

NoOfAnalyticalPortionsValue is a group term, which refers to all NoOfAnalyticalPortionsValue related fields. The term is allowed only in the SELECT-clause.

EnergyDistribution

The ComponentValue is used as a percentage of energy from the total energy of the food. This is possible only for the energy containing nutrients. The term is experimental.

Metainformation Related Rules

AvailableComponents

Term 'AvailableComponents' is a group term which refers to a similar element that ComponentList (see Component related rules). Implementation see the Web service:

- GetFCDBContent

AvailableFoods

Term 'AvailableFoods' is a group term which refers to a similar element that FoodList (see Food related rules). Implementation, see the Web service:

- GetFCDBContent

Content

Content is a group term, which refers to the Content-element of the FDTP. The term is allowed only in the SELECT-clause. Implementation, see the Web services:

- GetContentInformation
- GetFCDBContent

Count

Count is a group term, which refers to (group by) counts like e.g. Get Food Count. (Used because SELECT-clause is mandatory in the FDQL Sentence) related fields. The term is allowed only in the SELECT-clause.

Implementation, see the Web services:

- GetFoodCount
- GetFoodCountByProductType

SupportedOrderByTerms

SupportedSelectTerms is a group term referring to the listings of supported terms and entities in the ORDER BY-clause. Implementation, see the Web service:

- GetSupportedTerms

SupportedSelectTerms

SupportedSelectTerms is a group term referring to the listings of supported terms and entities in the SELECT-clause. Implementation, see the Web service:

- GetSupportedTerms

SupportedWhereTerms

SupportedSelectTerms is a group term referring to the listings of supported terms and entities in the WHERE-clause. Implementation, see the Web service:

- GetSupportedTerms

Metadata Transport Package (MDTP)

Metainformation (or metadata) is used for providing information about the information source i.e. FCDB. Compared with the FDTP, this is used for providing information about Web service like listing which terms are supported or which components are available. This information can be used by the user application or user application providers. Moreover, sometimes the food-oriented structure is not optimal for data which is not food-oriented and it is simpler to put that into a slightly different package without unnecessary bending of the structure or the rules of the FDTP.

Main structure of the MDTP

While the FDTP has strict rules and is highly standardized, the MDTP is a multi-purpose transport package. Its main structure is similar to the FDTP and even the first elements are completely the same:

- StandardVocabularies
- SenderInformation
- Content

The other five elements are optional and their usage depends on the Web service:

- FCDB_Describe
- ComponentList
- FoodList
- TermList
- Grouping

```
<EuroFIRMetaDataTransportPackage version="1.2" sentdate="2012-12-24">
  <StandardVocabularies/>
  <SenderInformation/>
  <Content/>
<FCDB_Describe/>
  <ComponentList/>
  <FoodList/>
  <TermList/>
  <Grouping/>
</EuroFIRMetaDataTransportPackage>
```

Grouping -structure

Many elements use a multi-purpose Grouping-structure. The structure consists of MainGroup element and Group element. The Group element has a label (usually a code of some kind) and value (like e.g. count). If the Grouping is nested, the Group is surrounded by one or more MainGroups. MainGroup-elements have only a label but they have no value. Labels have an optional reference (system) for binding the group to some classification scheme. All standard vocabularies used with these bindings MUST be declared in the standard vocabularies section.

```
<Grouping name="Grouping title">
  <MainGroup>
    <Label system="reference to main grouping">Some main group label</Label>
    <MainGroup>
      <Label system="reference to nested grouping">Some nested main group label</Label>
      <Group>
        <Label system="reference to grouping">Group label</Label>
        <Value>1234</Value>
      </Group>
    </MainGroup>
  </MainGroup>
</Grouping>
```

Food count example illustrates this grouping structure. First, the Grouping-element has a title "Food Count By product type". Then, the Grouping has two Group-elements: the Label, in this case a product type –code, and the Value finally presents the count. The system-attribute of the Label binds the group with the product type of the standard vocabularies.

```
<Grouping name="Food Count By Product type">
  <Group>
    <Label system="prodtype">A0792</Label>
    <Value>12</Value>
  </Group>
  <Group>
    <Label system="prodtype">A0791</Label>
    <Value>34</Value>
  </Group>
</Grouping>
```

FCDB_Describe

This element provides information about the FCDB like e.g. the number of Foods or Component Values (see Web service GetFCDBContent).

```
<FCDB_Describe>
  <Grouping name="Some FCDB description">
    <Group>
      <Label>Food</Label>
      <Value>1000</Value>
    </Group>
    <Group>
      <Label>Component</Label>
      <Value>20</Value>
    </Group>
    <Group>
      <Label>ComponentValue</Label>
      <Value>10000</Value>
    </Group>
  </Grouping>
</FCDB_Describe>
```

ComponentList

Components-element is similar to the Component Identifier element in the FDTP.

FoodList

Foods-element is similar to the Food Description element in the FDTP but without Recipe-element.

TermList

TermList-element is used for presenting different lists of terms like the supported WHERE-clause terms. The entityName-attribute links the term with the attribute.

```
<TermList name="Some list name">
  <Term entityName="Some entity name"/>
</TermList>
```

Example

```
<TermList name="Supported Where Terms">
  <Term entityName="Food">origfdcd</Term>
  <Term entityName="Food">FoodIdentifierLanguag</Term>
  <Term entityName="Food">FoodName</Term>
  <Term entityName="Component">ecompid</Term>
</TermList>
```

Grouping

Grouping-element is used for similar purposes than the SQL GROUP BY-clause. It uses the previously defined Grouping-structure

Web Services

GetComponentList

Use Case

6.5.3 and an alternative path to 6.5.3 defined in page 32 of the Background report.

Description

Produces a list of Components. Consists of Component Identifier -elements of the FDTP

Parameters

Name	Explanation
api_userid	User application identification key. See Web services authentication
api_permission	User profile permission, but currently used for identifying desired database country. See Web services authentication
fdql_sentence	See Food Data Query Language (FDQL).
version	Version of the Web service, this version 1.2.
api_signature	Message signature. See Web services authentication

Semantic rules

Only allowed SELECT-clause term is GetComponentList. WHERE-clause may include Component related conditions (see reserved words). No ORDER BY-clause. Because all standard vocabularies are excluded, only ecompid and (if exists) origcpd (see FDQL Data model).

Response

Response uses the Metadata Transport Package (MDTP).

Error messages

See Error Messages and Error Codes

GetContentInformation

Use Case

6.5.1 of the Background report.

Description

Retrieve FCDB version information defined as the Content element of the FDTP (All elements before Foods)

Parameters

Name	Explanation
api_userid	User application identification key. See Web services authentication
api_permission	User profile permission, but currently used for identifying desired database country. See Web services authentication
fdql_sentence	See Food Data Query Language (FDQL).
version	Version of the Web service, this version 1.2.
api_signature	Message signature. See Web services authentication

Example of the fdql_sentence

```
<FDQL_Sentence>
  <MetaData>
    <SchemaVersion>1.2</SchemaVersion>
    <Schema> EuroFIR_Web_Service_FDQL_Sentence_version_1_2.xsd</Schema>
  </MetaData>
  <SelectClause>
    <FieldName>Content</FieldName>
  </SelectClause>
</FDQL_Sentence>
```

Semantic rules

Only allowed SELECT-clause term is Content. No WHERE or ORDER BY-clause (see reserved words).

Response

Response uses the Metadata Transport Package (MDTP).

Error messages

See Error Messages and Error Codes

GetFCDBContent

Use Case

6.5.1 of the Background report.

Description

Retrieve FCDB version information and basic information of the FCDB content.

Parameters

Name	Explanation
api_userid	User application identification key. See Web services authentication
api_permission	User profile permission, but currently used for identifying desired database country. See Web services authentication
fdql_sentence	See Food Data Query Language (FDQL).
version	Version of the Web service, this version 1.2.
api_signature	Message signature. See Web services authentication

Example of the fdql_sentence

```
<FDQL_Sentence>
  <MetaData>
    <SchemaVersion>1.2</SchemaVersion>
    <Schema> EuroFIR_Web_Service_FDQL_Sentence_version_1_2.xsd</Schema>
  </MetaData>
  <SelectClause>
    <FieldName>Content</FieldName>
    <FieldName>AvailableComponents</FieldName>
  </SelectClause>
</FDQL_Sentence>
```

Semantic rules

Only allowed SELECT-clause terms are Content, AvailableFoods and AvailableComponents. No WHERE or GROUP BY-clause. (see reserved words).

Term	Explanation
Content	Produces a grouping with the number of available Foods, number of available Components and Component Values
AvailableFoods	Produces a FoodList of the available foods.
AvailableComponents	Produces a ComponentList of the available components.

Response

Response uses the Metadata Transport Package (MDTP).

Example

See Metadata Transport Package (MDTP).

Error messages

See Error Messages and Error Codes

GetFoodCount

Use Case

6.5.5.1 of the Background report.

Description

A count of available foods. WHERE-clause may include Food related conditions

Parameters

Name	Explanation
api_userid	User application identification key. See Web services authentication
api_permission	User profile permission, but currently used for identifying desired database country. See Web services authentication
fdql_sentence	See Food Data Query Language (FDQL).
version	Version of the Web service, this version 1.2.
api_signature	Message signature. See Web services authentication

Example of the fdql_sentence

The example gives a count of Foods which have an English name 'Avocado'

```
<FDQL_Sentence>
  <MetaData>
    <SchemaVersion>1.2</SchemaVersion>
    <Schema> EuroFIR_Web_Service_FDQL_Sentence_version_1_2.xsd</Schema>
  </MetaData>
  <SelectClause>
    <FieldName>Count</FieldName>
  </SelectClause>
  <WhereClause>
    <Condition xsi:type="T_CommonCondition" logicalOperator="AND">
      <NameConditionField language="en">
        <FieldName>FoodName</FieldName>
      </NameConditionField>
      <ConditionOperator>LIKE</ConditionOperator>
      <ConditionValue>Avocado</ConditionValue>
    </Condition>
  </WhereClause>
</FDQL_Sentence>
```

Semantic rules

Only allowed SELECT-clause terms is count. The WHERE-clause allows only Food related terms (see reserved words). No ORDER BY-clause

Response

Response uses the Metadata Transport Package (MDTP).

Example

Note that StandardVocabularies and SenderInformation have been simplified.

```
<EuroFIRMetaDataTableTransportPackage version="1.2" sentdate="2012-12-24">
  <StandardVocabularies/>
  <SenderInformation/>
  <Content datasetcreated="2012-12-24" language="tih" acquisitiontype="F">
  </Content>
  <Grouping name="Food count" type="Level_1">
    <Group>
      <Label>Foods</Label>
      <Value>1234</Value>
    </Group>
  </Grouping>
</EuroFIRMetaDataTableTransportPackage>
```

Error messages

See Error Messages and Error Codes

GetFoodCountByProductType

Use Case

6.5.5.2 of the Background report.

Description

A count grouped of available foods. EuroFIR food classification [A0777] will be used for grouping factor. WHERE-clause may include Food related conditions

Parameters

Name	Explanation
api_userid	User application identification key. See Web services authentication
api_permission	User profile permission, but currently used for identifying desired database country. See Web services authentication
fdql_sentence	See Food Data Query Language (FDQL).
version	Version of the Web service, this version 1.2.
api_signature	Message signature. See Web services authentication

Example of the fdql_sentence

This example gives a group by count of Foods which have prodtype='A0790'

```
<FDQL_Sentence>
  <MetaData>
    <SchemaVersion>1.2</SchemaVersion>
    <Schema>EuroFIR_Web_Service_FDQL_Sentence_version_1_2.xsd</Schema>
  </MetaData>
  <SelectClause>
    <FieldName>Content</FieldName>
  </SelectClause>
  <WhereClause>
    <Condition xsi:type="T_CommonCondition" logicalOperator="AND">
      <ClassificationConditionField searchScope="BT">
        <FieldName>FoodIdentifierLanguag</FieldName>
      </ClassificationConditionField>
      <ConditionOperator>=</ConditionOperator>
      <ConditionValue>A0790</ConditionValue>
    </Condition>
  </WhereClause>
</FDQL_Sentence>
```

Semantic rules

Only allowed SELECT-clause term is count. WHERE-clause allows only Food related terms (see reserved words). No ORDER BY-clause

Response

Response uses the Metadata Transport Package (MDTP).

Example

see Metadata Transport Package (MDTP).

Error messages

See Error Messages and Error Codes

GetFoodInformation

Use Case

6.2.2, 6.2.3, 6.2.4, 6.3.1, 6.3.2, 6.3.3 and 6.3.4 of the Background report.

Description

This is the main service for retrieving food information. Retrieve foods, components and component values using FDQL for the selection options.

Parameters

Name	Explanation
api_userid	User application identification key. See Web services authentication
api_permission	User profile permission, but currently used for identifying desired database country. See Web services authentication
fdql_sentence	See Food Data Query Language (FDQL).
version	Version of the Web service, this version 1.2.
api_signature	Message signature. See Web services authentication

Example of the fdql_sentence

The example gives all FDTP minimum requirements based information of Foods which have an English name 'Avocado' and VITC from Components (and Component Values).

```
<FDQL_Sentence>
  <MetaData>
    <SchemaVersion>1.2</SchemaVersion>
    <Schema>EuroFIR_Web_Service_FDQL_Sentence_version_1_2.xsd</Schema>
  </MetaData>
  <SelectClause>
    <FieldName>FoodAllMinimum</FieldName>
    <FieldName>ComponentAllMinimum</FieldName>
    <FieldName>ComponentValueAllMinimum</FieldName>
  </SelectClause>
  <WhereClause>
    <Condition xsi:type="T_CommonCondition" logicalOperator="AND">
      <NameConditionField language="en">
        <FieldName>FoodName</FieldName>
      </NameConditionField>
      <ConditionOperator>LIKE</ConditionOperator>
      <ConditionValue>Avocado</ConditionValue>
    </Condition>
    <Condition xsi:type="T_CommonCondition" logicalOperator="AND">
      <ClassificationConditionField searchScope="NT">
        <FieldName>ecompid</FieldName>
      </ClassificationConditionField>
      <ConditionOperator>=</ConditionOperator>
      <ConditionValue>VITC</ConditionValue>
    </Condition>
  </WhereClause>
</FDQL_Sentence>
```

Semantic rules

A full set of fields available in SELECT, WHERE and ORDER BY-clauses (see restrictions and reserved words). Currently no Component Value-related conditions are allowed in the WHERE-clause

Response

Response uses the FDTP.

Error messages

See Error Messages and Error Codes

GetFoodList

Use Case

6.5.2 and an alternative path to 6.5.2 defined in page 32 of the Background report.

Description

Produces a list of Foods: FDTP Food Description-element without Recipe-element.

Parameters

Name	Explanation
api_userid	User application identification key. See Web services authentication
api_permission	User profile permission, but currently used for identifying desired database country. See Web services authentication
fdql_sentence	See Food Data Query Language (FDQL).
version	Version of the Web service, this version 1.2.
api_signature	Message signature. See Web services authentication

Semantic rules

Only allowed SELECT-clause term is FoodtList. No ORDER BY-clause. WHERE-clause allows only Food related terms (see reserved words). FoodList includes all Food related mandatory information of the FDTP Food-element (not Components or Component Value and their subordinate entities).

Response

Response uses the Metadata Transport Package (MDTP).

Error messages

See Error Messages and Error Codes

GetSupportedTerms

Description

Produces a list of terms which are supported by the Partner Web service.

Parameters

Name	Explanation
api_userid	User application identification key. See Web services authentication
api_permission	User profile permission, but currently used for identifying desired database country. See Web services authentication
fdql_sentence	See Food Data Query Language (FDQL).
version	Version of the Web service, this version 1.2.
api_signature	Message signature. See Web services authentication

Semantic rules

Only allowed SELECT-clause terms are supported (see reserved words):

- SupportedSelectTerms – terms supported in FDQL SELECT-clause
- SupportedWhereTerms – terms supported in FDQL WHERE-clause
- SupportedOrderByTerms – terms supported in FDQL ORDER BY-clause

No WHERE/ORDER BY-clause.

Response

Response uses the Metadata Transport Package (MDTP).

Example

Note that StandardVocabularies, SenderInformation and Content have been simplified

```
EuroFIRMetaDataTransportPackage version="1.2" sentdate="2012-12-24">
  <StandardVocabularies/>
  <SenderInformation/>
  <Content/>
  <TermList name="Supported Where Terms">
    <Term entityName="Food">origfdcd</Term>
    <Term entityName="Food">FoodIdentifierLanguag</Term>
    <Term entityName="Food">FoodName</Term>
    <Term entityName="Component">ecompid</Term>
  </TermList>
</EuroFIRMetaDataTransportPackage>
```

Error messages

See Error Messages and Error Codes

Error Messages and Error Codes

It is a common fact that there are no systems without errors. In such case the Web service SHALL send a proper error message and give user application as much information as possible about what went wrong. These error codes and error messages are, however, the minimum reporting when an error occurs and using them is REQUIRED.

The error message response includes first the SOAP standard fault code in the faultcode-element. Then our error message is in the faultstring-element. These are REQUIRED. The Web service MAY give additional information in the detail-element: the sub-element 'errorcode' is reserved for our error code and the sub-element 'reason' for a detailed error description with a free text (see Web Service Response).

The error list covers the most probable errors. Some of the error codes are mentioned in the other parts of the specification and some are not. All of these error codes, however, may be used. Some of the error codes are quite general and some are very specific.

Table 7. Errors when the request is received.

Error message		Error code
Request receive error		E1000
	Consumer error	E1010:
		Formatting error
		Server not responding
		Unexpected response
	Server error	E1020
		Unknown request format
		Parameter mismatch with the service
		Non-existing service

Table 8. Errors during the authentication

Error message			Error code
Request authentication error			E2000
	Pre-authentication error		E2010
		No user application identification key (api_userid)	E2011
		No message signature (api_signature)	E2012
		No response from authentication server	E2013
	Authentication error		E2020
		Invalid user application identification key (api_userid)	E2021
		Invalid message signature checking	E2022
	Post-authentication error		E2030
		Access denied to user	E2031
		Undefined user application permission key (api_permission)	E2032

Table 9. Errors during the FDQL sentence processing and query

Error message		Error code
FDQL sentence processing error		E3000
	FDQL Pre-processing error	E3010
		Error parsing query parameters
		E3011
		FDQL validation error
		E3012
		FDQL translation error
		E3013
		FDQL unknown field error
		E3014
		FDQL empty select fields
		E3015
		FDQL non-existing field error
		E3016
		FDQL field not supported in the FDQL
		E3017
		FDQL select field not supported in the FDQL
		E3018
		FDQL where field not supported in the FDQL
		E3019
		FDQL order by field not supported in the FDQL
		E3020
		FDQL field translation error
		E3021
		FDQL select field translation error
		E3022
		FDQL where field translation error
		E3023
		FDQL order by field translation error
		E3024
		FDQL field mismatch error
		E3025
		FDQL select field mismatch error
		E3026
		FDQL where field mismatch error
		E3027
		FDQL order by field mismatch error
		E3028
		FDQL language code not supported
		E3029
		FDQL standard vocabulary not supported
		E3030
		FDQL standard vocabulary code not supported
		E3031
		FDQL language code not supported
		E3032
		FDQL Language code not supported
		E3033
		FDQL Language facet not supported
		E3034
		Permission limitations
		E3098
		No response from data source
		E3099
	FDQL Data processing error	E3100
		Data source reported error - ACL Limitations
		E3101
		Data source reported error - query error
		E3102
		Data source reported error - result/viewing error
		E3103
	FDQL Post data process/result formulation error	E3130
		Error in parsing dataset
		E3131

Table 10. Errors during the return of the data and unknown error

Error message			Error code
Data return error			E4000
	Response formulation error		E4010
		Error in formulating response - missing data	E4011
		Error in XML creation	E4012
	Clean up error		E4020
Unknown error			E5000

References

- (1) EuroFIR AISBL. EuroFIR - Your source of food information; Available at: <http://www.eurofir.org/>. Accessed 2/4, 2015.
- (2) W3C. SOAP Version 1.2 Part 1: Messaging Framework (Second Edition). 2007; Available at: <http://www.w3.org/TR/soap12-part1/>. Accessed 2/4, 2015.
- (3) Møller A, Unwin ID, Ireland J, Roe MA, Becker W, Colombani P. The EuroFIR Thesauri 2008 - EuroFIR Technical Report D1.8.22; Available at: http://www.eurofir.org/wp-content/uploads/TechWeb%20Downloads/Thesauri/EuroFIR_Thesauri_2008.pdf. Accessed 2/4, 2015.
- (4) Becker W, Unwin I, Ireland J, Møller A. Proposal for structure and detail of a EuroFIR standard on food composition data I: Description of the standard. 2007.
- (5) Becker W, Møller A, Ireland J, Roe M, Unwin I, Pakkala H. Proposal for structure and detail of a EuroFIR standard on food composition data II: Technical Annex. 2008.
- (6) Møller A, Christensen T. EuroFIR Web Services - EuroFIR Food Data Transport Package, Version 1.4. 2012.
- (7) Møller A, Ireland J. LanguaL 2012 – The LanguaL Thesaurus. 2012 http://www.langual.org/langual_Downloads.asp.
- (8) Pakkala H, Christensen T, Gunnarsson Í, Kadvan A, Keshet B, Korhonen T, et al. EuroFIR Web Services: Background Report. 2008.
- (9) Bradner S. RFC 2119 Key words for use in RFCs to Indicate Requirement Levels. Best Current Practice. 1997; Available at: <http://tools.ietf.org/html/rfc2119>. Accessed 2/4, 2015
- (10) International Organization for Standardization. ISO 639:1988 (E/F). Code for the Representation of Names of Languages. 1988.
- (11) International Organization for Standardization. ISO 3166-1 Codes for the representation of names of countries and their subdivisions. 1997.
- (12) Alvestrand H. RFC 3066: Tags for the Identification of Languages. 2001; Available at: <http://www.ietf.org/rfc/rfc3066.txt>. Accessed 2/4, 2015.
- (13) Yergeau F. RFC 3629. UTF-8, a transformation format of ISO 10646. 2003; Available at: <http://tools.ietf.org/html/rfc3629>. Accessed 2/4, 2015.
- (14) W3C. Extensible Markup Language (XML) 1.0 (Fifth Edition). 2008; Available at: <http://www.w3.org/TR/REC-xml/>. Accessed 2/4, 2015.

(15) Eastlake D, Jones P. RFC 3174 - US Secure Hash Algorithm 1 (SHA1). 2001; Available at: <http://www.ietf.org/rfc/rfc3174.txt>. Accessed 2/4, 2015.