

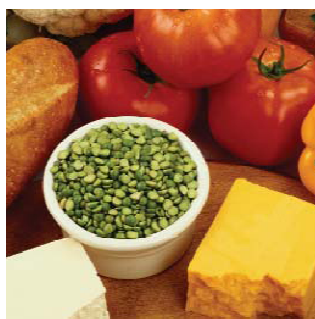
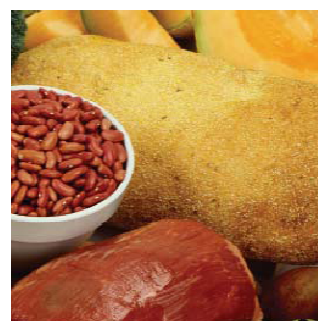
This work was completed on behalf of the European Food Information Resource (EuroFIR) Consortium and funded under the EU 6th Framework Food Quality and Safety thematic priority. Contract FOOD-CT-2005-513944.

## EuroFIR Web Services

Specification of request-response message exchange patterns  
Version 1.0

Heikki Pakkala, Tue Christensen, Ívar Gunnarsson,  
Agnes Kadvan, Benny Keshet, Tommi Korhonen,  
Ignacio Martínez de Victoria, Anders Møller, Karl Presser,  
Paolo Colombani, Erik Nørby

EuroFIR Technical Report D1.8.29



**Disclaimer**

This work was completed on behalf of the European Food Information Resource (EuroFIR) Consortium and funded under the EU 6<sup>th</sup> Framework Quality and Safety Programme, project number FP6-513944.

EuroFIR, the world leading European Network of Excellence on Food Composition Databank systems (<http://www.eurfir.net/>) is a partnership between 49 universities, research institutes and small-to-medium sized enterprises (SMEs) from 26 countries. EuroFIR aims to develop and integrate a comprehensive, coherent and validated databank providing a single, authoritative source of food composition data for Europe.

**EuroFIR Project Management Office  
Institute of Food Research, Norwich Research Park  
Norwich, Norfolk, NR4 7UA, UK**

**EuroFIR Web Services**  
**SPECIFICATION OF**  
**REQUEST-RESPONSE MESSAGE EXCHANGE PATTERNS**  
**VERSION 1.0**

**HEIKKI PAKKALA**  
**TUE CHRISTENSEN**  
**ÍVAR GUNNARSSON**  
**AGNES KADVAN**  
**BENNY KESHET**  
**TOMMI KORHONEN**  
**IGNACIO MARTÍNEZ DE VICTORIA**  
**ANDERS MØLLER**  
**KARL PRESSER**  
**PAOLO COLOMBANI**  
**ERIK NØRBY**

## LEGAL NOTICE

Neither the EuroFIR Consortium nor any person acting on behalf of the EuroFIR Consortium is responsible for the use which might be made of the following information.

Information on the EuroFIR project is available on the Internet. It can be accessed through the EuroFIR server (<http://www.eurofir.net>).

Cataloguing information:

**EuroFIR Web Services - Specification Of Request-Response Message Exchange Patterns, Version 1.0**

*Heikki Pakkala, Tue Christensen, Ívar Gunnarsson, Agnes Kadvan, Benny Keshet, Tommi Korhonen, Ignacio Martínez de Victoria, Anders Møller, Karl Presser, Paolo Colombani, Erik Nørby*

Denmark: Danish Food Information

2008 - 101 pp. - 21 x 29.7 cm.

ISBN 978-87-92125-12-5

EAN 9788792125125

© The EuroFIR Consortium, 2008

Reproduction is authorised provided the source is acknowledged.

*Printed in Denmark.*

# Content

Introduction .....	6
Objective .....	6
Background .....	6
Terminology .....	7
General Terms .....	8
Process Flow .....	9
Web Service Request .....	10
Web Service Response .....	11
Normal Response .....	11
Error Message .....	11
Error Message .....	12
Web Services Authentication .....	13
Food Data Query Language (FDQL) .....	16
FDQL Data Model .....	16
FDQL Data Model Restrictions .....	19
FDQL Reserved Words .....	20
FDQL Implementation .....	23
FDQL Sentence Structure in XML .....	23
Example of the FDQL Sentence .....	23
Semantic Rules for the FDQL Sentence Translation .....	25
General Interpretation Rules .....	25
Joins Between the Entities .....	26
Validation of the FDQL Sentence .....	28
WHERE-clause Evaluation .....	29
Ordering With the ORDER BY -clause .....	30
Food Related Semantic Rules .....	30
Component Related Rules .....	32
Component Value Related Rules .....	33
Metainformation Related Rules .....	35
Metadata Transport Package (MDTP) .....	37
Web Services .....	40
GetComponentList .....	40
GetContentInformation .....	41
GetFCDBContent .....	42
GetFoodCount .....	44
GetFoodCountByProductType .....	45
GetFoodInformation .....	48
GetFoodList .....	50
GetSupportedTerms .....	51
Error Messages and Error Codes .....	53
Appendix 1 XML Schemas .....	59
EuroFIR_Web_Service_FDQL_Sentence_version_1_0.xsd .....	59
EuroFIR_Unit_Thesaurus_version_1_0.xsd .....	70
EuroFIR_Matrix_Unit_Thesaurus_version_1_0.xsd .....	71
EuroFIR_Component_Thesaurus_version_1_0.xsd .....	71
EuroFIR_Web_Service_FDQL_Reserved_Words_version_1_0.xsd .....	80
Appendix 2 EuroFIR Web Service WSDL .....	87
Documentation .....	87
Source .....	97

## Introduction

European Food Information Resource Network (EuroFIR), the world-leading European Network of Excellence on Food Composition Databank systems ([www.eurofir.net](http://www.eurofir.net)) is a partnership between 48 universities, research institutes and small-to-medium sized enterprises (SMEs) from 27 countries [1]. EuroFIR aims to develop and integrate a comprehensive, coherent and validated databank providing a single authoritative source of food composition data for Europe. Currently, EuroFIR is in the transition phase from a project to a non-profit organization. In this report "EuroFIR" refers to the current project (2005-2009) and "EuroFIR AISBL" (Association internationale sans but lucratif) refers to the new organization succeeding the EuroFIR project from 2009.

This report was completed on behalf of the EuroFIR Consortium and funded under the EU 6th Framework Food Quality and Safety Programme, project number FOOD-CT-2005-513944.

## Objective

This specification defines the EuroFIR Web Services, version 1.0. 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.

## 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 [2].
- Proposal for structure and detail of a EuroFIR standard on food composition data: II: Technical annex[3]; referenced in this document as "Technical Annex".
- EuroFIR Web Services - Food Data Transport Package, Version 1.3. [4]; referenced in this document as "FDTP"
- LanguaL Food Description 2008 Thesaurus [5-8]; referenced in this document as "LanguaL"
- EuroFIR Web Services: Background report[9]; 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, making this possible guides our interpretation. So, the 'interpretation order' for the implementation of these EuroFIR Web Services is:

1. this document
2. FDTP
3. Technical Annex
4. other above mentioned documents

## 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[10].

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

<b>Thesaurus</b>	<b>Version</b>	<b>Date</b>	<b>Reference</b>
EuroFIR Acquisition Type Thesaurus	1.0	2007-11-04	[11]
EuroFIR Component Thesaurus	1.0	2008-03-12	[12]
EuroFIR Value Type Thesaurus	1.0	2007-11-04	[13]
EuroFIR Unit Thesaurus	1.0	2007-11-04	[14]
EuroFIR Matrix Unit Thesaurus	1.0	2007-11-04	[15]
EuroFIR Method Type Thesaurus	1.0	2007-11-04	[16]
EuroFIR Method Indicator Thesaurus	1.0	2007-11-04	[17]
EuroFIR Reference Type Thesaurus	1.0	2007-11-04	[18]
LanguaL Thesaurus	2008	2008	[6]

Table 2. Other naming standards

<b>Naming standard</b>	<b>Date</b>	<b>Reference</b>
ISO 639. Code for the representation of the names of languages	1988	[11 see also Technical Annex]
ISO 3166-1 Codes for the representation of names of countries and their subdivisions	1997	[19 see also Technical Annex]
RFC 3066: Tags for the Identification of Languages. (XML language tags)	2001	[20]

## **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 EuroFIR partners (or 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. The most important search tool will probably be the EuroFIR eSearch facility [21]. 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). These persons are often food compilers.

### **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 Web service providers. This specification assumes that these FCDBs meet the requirements of the Technical Annex.

### **FDTP**

Food Data Transportation Package used in the data interchange [4]

## **MDTP**

Metadata Transportation Package is an experimental transportation package used for providing information about the Partner Web service and its FCDB. (See chapter Metadata Transportation 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” [22])

## **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"[22].

The request

- MUST use either the UTF-8 [23] or UTF-16 (Unicode) [24] encoding.
- MUST be a well-formed (XML 1.0) document [25]
- MUST contain method element for Web service identification
- MUST contain parameters for authentication.

The request in the envelope is posted to a URL. Each Web service provider is RECOMMENDED to give the SOAP Endpoint URL as:

*XX/eurofirservices/soap/*

*where XX is http://SomePartnerSpecificURL*

```
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
  <env:Body>
    <methodname>
      <parametername>value</parametername>
    </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. GetFoodInformation) Each request parameter SHALL be a single child of that. The method element identifies the Web Service where the request is delegated to.

## Web Service Response

Web service responses SHALL use SOAP "envelopes" [22]

The response

- MUST use either the UTF-8 [23] or UTF-16 (Unicode) [24] encoding. UTF-8 is RECOMMENDED as FDTP uses that as default encoding in the future.
- MUST be a well-formed (XML 1.0) document [25]

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

```
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
  <env:Body>
    <EuroFIRServiceResponse>
      <EuroFIRFoodDataTransportPackage>
        put your data here
      </EuroFIRFoodDataTransportPackage>
    </EuroFIRServiceResponse>
  </env:Body>
</env:Envelope>
```

### Metadata Transfer Package (MDTP)

```
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
  <env:Body>
    <EuroFIRServiceResponse>
      <EuroFIRMetaDataTransportPackage>
        put your data here
      </EuroFIRMetaDataTransportPackage>
    </EuroFIRServiceResponse>
  </env:Body>
</env:Envelope>
```

## Error Message

```
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
  <env:Body>
    <env:Fault>
      <faultcode>SOAP FAULT CODE</faultcode>
      <faultstring>Some Web service error message</faultstring>
      <detail>
        <EuroFIRServiceFault>
          <errorcode>Some Web service error code</errorcode>
          <reason>Some error description</reason>
        </EuroFIRServiceFault>
      </detail>
    </env:Fault>
  </env:Body>
</env:Envelope>
```

### **faultcode**

SOAP standard fault codes. REQUIRED.

### **faultstring**

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

### **detail**

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

## 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. Each user application may have different access profiles with different permissions. However, all actions of the Web services using these different permissions must be defined in this specification. This makes possible for the user application provider, to give different profiles to different end users. The user application provider is responsible for controlling all user access management including these profile based rights (see also Figure 1).

This authentication model has been influenced by those of the Amazon Web Services [26] and the Flickr Services Authentication API [27].

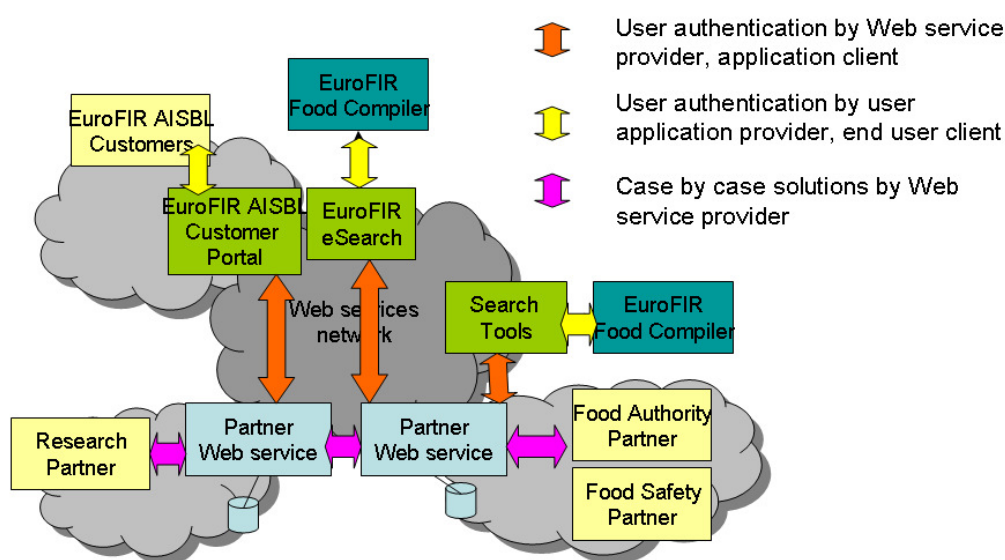


Figure 1. Web services authentication scheme

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

### **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'.

### **Permission**

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

- Currently, the only permission is 'unlimited' and no access profiles have been defined.
- 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 MD5 Message-Digest Algorithm (MD5) [28, 29]
- 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 md5()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 **MUST** be 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).

## **Checking procedure**

- Web service authentication reads the user application identification key and checks if that is among the allowed user application identifier keys. Otherwise the access is revoked.
- If the key exists, the Web service authentication finds the user application secret key matching the user application identification key
- Then the Web service authentication calculates the signature using the request parameters (except the signature parameter itself) using the same procedure which is used for the calculation
- Then the Web service authentication compares this calculated signature with the signature in the request. If they match, access is granted. Otherwise the access is revoked.

## **Risk management of the identification keys**

The Web service provider **MUST** store the user application identification keys and secret key pairs safely so that they **SHALL NOT** be accessed by Web service users or non-authorized users. Moreover, 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.

## **Errors**

See Error messages and error codes

## 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)

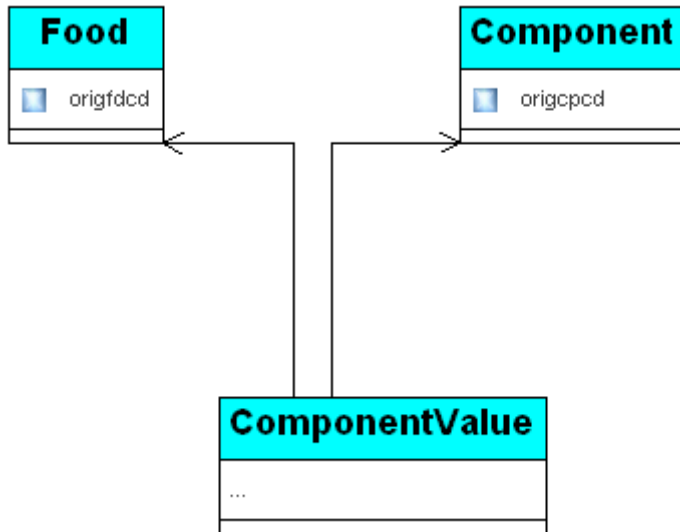


Figure 2. Main entities.

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 that in a sense 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).

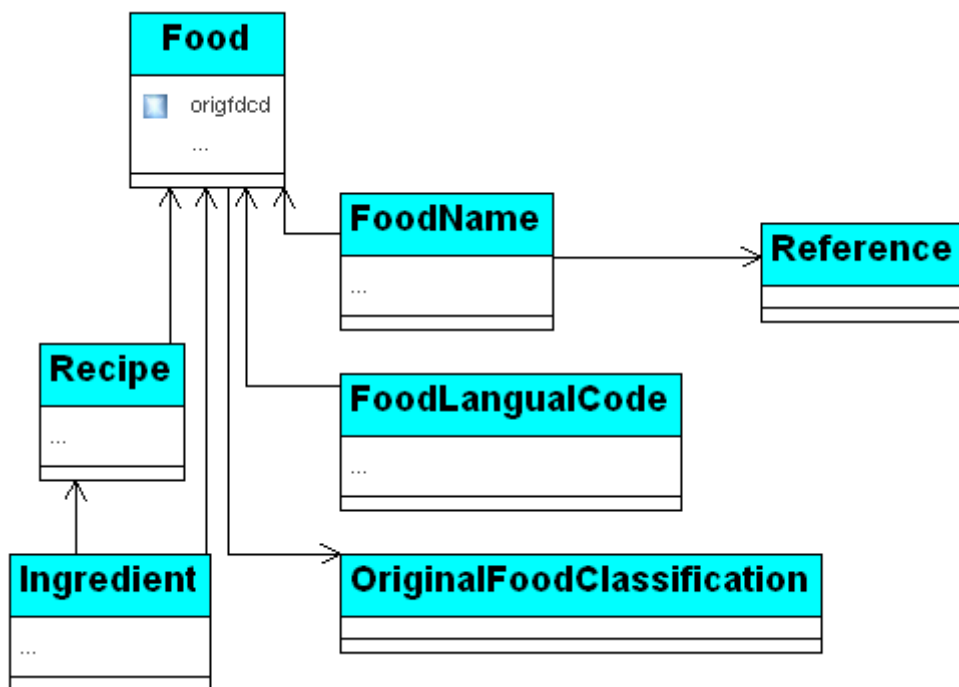


Figure 3. Food related entities.

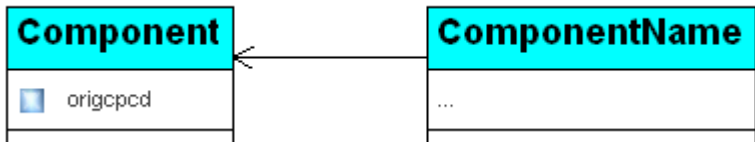


Figure 4. Component related entities.

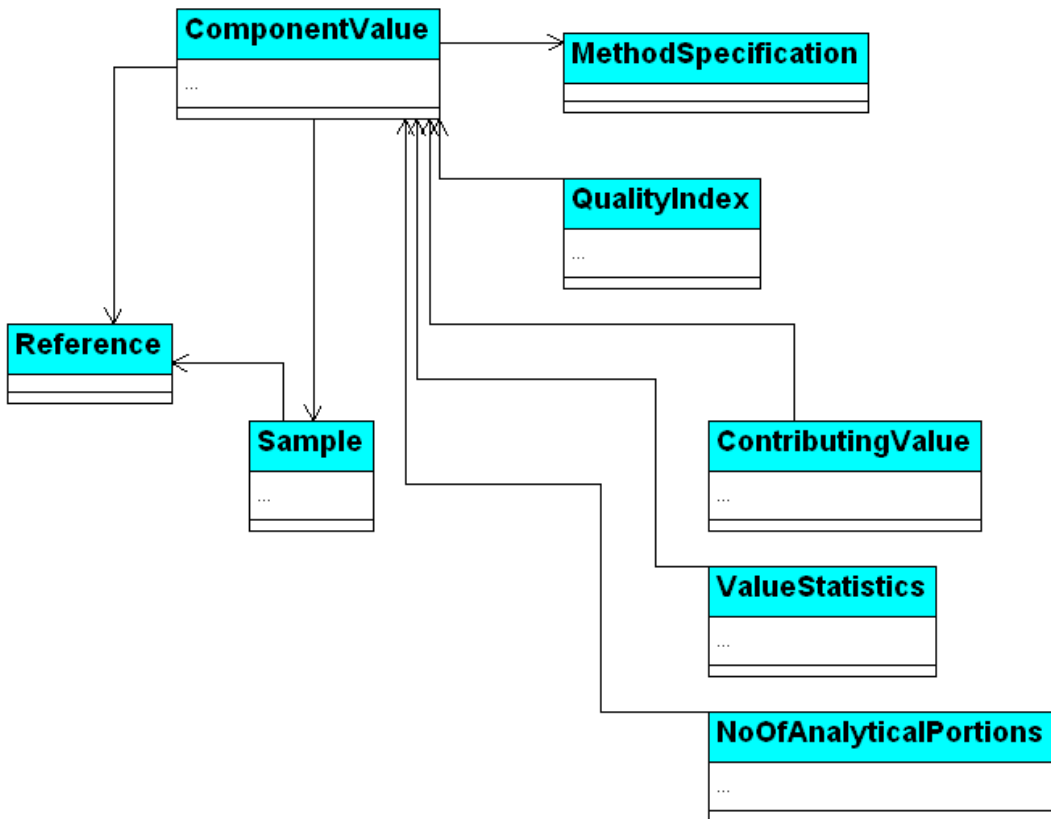


Figure 5. Component Value related entities.

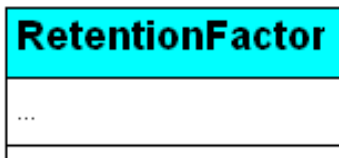


Figure 5. Assisting entities.

We use a term 'field' referring to an element which is commonly a column in a relational data base. 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 standard food classifications and general food description (in the Technical Annex). The Technical Annex defines several property identifiers (like e.g.'COOKMETH') for them and in the FCDBs they may be 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. Also the FDTP groups them under the FoodClasses element. Only non-LanguaL classification is the original local food classification.

### **Retention factor not used**

Retention factor is not used in this version.

### **Limitation of use in different Web services**

Different Web services may not use all the entities. See the semantic rules.

### **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, and it is not possible to use e.g. Component Value attributes in the WHERE-clause. See the FDQL reserved words.

### **Considerable differences between different Partner Web services**

Different Partner Web services may not use all the entities or fields; many of them are optional. See the FDTP about the current minimum requirements

## 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
FoodLanguagCode	FoodIdentifierLanguag				WHERE-clause defines the search scope (BT/NT), ClassificationConditionField
FoodName	FoodName				NameConditionField. The language is defined by the language attribute
OriginalFoodClassification	origgpcd				ClassificationConditionField
Recipe	Recipe				
Ingredient	Ingredient				
Food	FoodAll				Group term. See Semantic rules
Food	FoodAllMandatory				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				ClassificationConditionField, currently NT=BT (Currently EuroFIR Component Thesaurus version 1.0 does not have any hierarchy.)
Component	origcpcd				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 indeces. 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

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 MAY 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 [23] or UTF-16 (Unicode) [24] encoding. UTF-8 is RECOMMENDED as FDTP uses that as default encoding in the future.
- MUST be a well-formed (XML 1.0) document [25]

## FDQL Sentence Structure in XML

The XML schema in the Appendix 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. Mandatory.

### WhereClause

WhereClause may restrict the retrieved information content. Optional.

### OrderByClause

WhereClause may order the retrieved information content. Optional.

## 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 in the Appendix 1

```

<FDQL_Sentence
  <MetaData>
    <SchemaVersion>1.0</SchemaVersion>
    <Schema> EuroFIR_Web_Service_FDQL_Sentence_version_1_0</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 xml:lang="en">
        <FieldName>FoodName</FieldName>
      </NameConditionField>
      <ConditionOperator>LIKE</ConditionOperator>
      <ConditionValue>Somethi%</ConditionValue>
    </Condition>
    <Condition xsi:type="T_InCondition" logicalOperator="OR">
      <ClassificationConditionField searchScope="BT">
        <FieldName>FoodIdentifierLangual</FieldName>
      </ClassificationConditionField>
      <ConditionOperator>IN</ConditionOperator>
      <ConditionValue>A1234</ConditionValue>
      <ConditionValue>A3456</ConditionValue>
      <ConditionValue>G1234</ConditionValue>
    </Condition>
    <Condition xsi:type="T_CommonCondition" logicalOperator="AND">
      <NameConditionField xml:lang="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.

### No subordinate entity without the main entity – inclusion principle

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.

### Group term takes everything available from the entity – supplement principle

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 allways 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 (currently not possible), 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 (currently not available), 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 MUST be translated to empty elements (or attributes) when the FDTP is compiled i.e NULLs are treated as empty.

## **Validation of the FDQL Sentence**

The XML schema defines the FDQL sentence structure (see Appendix 1). There are also other XML schemas defining all terms and even things like units, matrix units and Component identifiers (see Appendix 1). They also 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 can be used for validation and using this XML schema validation is RECOMMENDED. If the schema validation is not used, the Web Service MUST use some other methods for validating the terms in the FDQL sentence and the correctness of the FDQL sentence. In any case, 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 schemas 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 MUST be interrupted and an error message is sent (See Error messages and error codes). Moreover, the current XML schemas 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 is REQUIRED to interrupt 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 MUST 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 evaluation 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. Allowed only in the SELECT-clause. Implementation is RECOMMENDED. If not implemented, result SHALL be the same as "FoodAllMandatory".

### 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). Allowed only in the SELECT-clause. Implementation is REQUIRED.

### 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. Implementation is REQUIRED.

### 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 hierarchial. A broader term always yields results to all narrower terms as well. Implementing the LanguaL search using the searchScope is REQUIRED.

## **EuroFIR Food Classification**

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

## **OriginalFoodClassification**

Term "origgpcd" 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. The implementation of the search with the "origgpcd" and with searchScope is RECOMMENDABLE. So, even if the searchScope attribute is mandatory in the FDQL sentence it may not have any effect.

## **FoodNames**

FDQL sentence uses FoodNames with a mandatory language attribute. Implementing this is REQUIRED. The FDQL sentence uses xml:lang attribute (see RFC 3066) which is equal with the FDTP convention (ISO 639 2 character code for language plus an optional 2 character standard ISO country code). Language "tx" is reserved for scientific names. The existence of the xml:lang 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. Also, as stated in the FDTP, including synonyms in the FDTP is optional.

## **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. As Recipe is not mandatory in the FDTP, it may not exist in the FCDB.

## **FoodList**

Term "FoodList" is a group term, which refers to the combination of origfdcd and FoodName. The term is used only in the Web Service Get Food List.

## **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. Allowed only in the SELECT-clause. Implementation is RECOMMENDED. If not implemented, result SHALL be the same as "ComponentAllMandatory".

### **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. Allowed only in the SELECT-clause. Implementation is REQUIRED.

### **ComponentAllMinimum**

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

### **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 currently this term SHALL NOT be 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). If it exists the implementation of the search is REQUIRED.

### **ecompid**

Term "ecompid" refers to the EuroFIR Component Thesaurus. It is used allways with a searchScope attribute: Broader term (BT)/ Narrower term (NT). As the EuroFIR Component Thesaurus version 1.0 has only one level, it the

NT attribute and BT MUST give the same result. Implementation is REQUIRED. It is also RECOMMENDED to implement the Web Service so, that full search with NT/BT could be implemented when the several levels in the Component Thesaurus will be available.

### **ComponentList**

Term "ComponentList" is a group term, which refers to the combination of ecompid and OriginalComponentName. 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. Implementation is RECOMMENDED. If not implemented, result SHALL be the same as "ComponentValueAllMandatory".

### **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. Allowed only in the SELECT-clause. Implementation is REQUIRED.

### **ComponentValueAllMinimum**

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

### **QualityIndex**

QualityIndex is a group term, which refers to all QualityIndex related fields. Allowed only in the SELECT-clause. Implementation is OPTIONAL.

### **MethodSpecification**

MethodSpecification is a group term, which refers to all MethodSpecification related fields. Allowed only in the SELECT-clause. Implementation is OPTIONAL.

### **QualityIndex**

QualityIndex is a group term, which refers to all QualityIndex related fields. Allowed only in the SELECT-clause. Implementation is OPTIONAL.

### **Sample**

Sample is a group term, which refers to all Sample related fields. Allowed only in the SELECT-clause. Implementation is OPTIONAL.

### **ContributingValue**

ContributingValue is a group term, which refers to all ContributingValue related fields. Allowed only in the SELECT-clause. Implementation is OPTIONAL.

### **ValueStatistics**

ValueStatistics is a group term, which refers to all ValueStatistics related fields. Allowed only in the SELECT-clause. Implementation is OPTIONAL.

### **ValueReference**

ValueReference is a group term, which refers to all ValueReference related fields. Allowed only in the SELECT-clause. Implementation is OPTIONAL.

### **MethodReference**

MethodReference is a group term, which refers to all MethodReference related fields. Allowed only in the SELECT-clause. Implementation is OPTIONAL.

### **NoOfAnalyticalPortionsValue**

NoOfAnalyticalPortionsValue is a group term, which refers to all NoOfAnalyticalPortionsValue related fields. Allowed only in the SELECT-clause. Implementation is OPTIONAL.

## **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. 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 information is not used for data interchange like transferring data from one FCDB to another: metainformation is used for providing information about Parter 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. For these reasons, we introduce a new transport package: Metadata Transport Package (MDTP). All Web services using the MDTP are optional.

While the FDTP has strict rules and is highly standardized, the MDTP is an experimental 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
- Components
- Foods
- TermList
- Grouping

### Main structure of the MDTP

```
<EuroFIRMetaDataTableTransportPackage version="1.0" sentdate="2008-12-24">
  <StandardVocabularies/>
  <SenderInformation/>
  <Content/>
  <FCDB_Describe/>
  <Components/>
  <Foods/>
  <TermList/>
  <Grouping/>
</EuroFIRMetaDataTableTransportPackage>
```

Many element use a multi-purpose GroupElement-structure where a level-attribute defines the nesting level (currently one).

```
<GroupElement>
  <GroupLabel level="1" reference="something"/>
  <GroupValue/>
</GroupElement>
```

For each level there is a GroupLabel element (usually a code of some kind) and a GroupValue-element for presenting e.g. the count. Then there is a reference-attribute linking the GroupLabel (i.e. the code). This GroupElement-structure is repeated if needed inside a Grouping-element. This Grouping-element has a name-attribute for presenting the grouping title. It has also a type-attribute for a potential XML-schema defining the nesting level.

```
<Grouping name="Some title" type="Level_1">
  <GroupElement/>
  <GroupElement/>
</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 GroupElements:the GroupLabel, in this case, a product type –code and the GroupValue finally presents the count.

```
<Grouping name="Food Count By Product type" type="Level_1">
  <GroupElement>
    <GroupLabel level="1" reference="prodtype">A0792</GroupLabel>
    <GroupValue>12</GroupValue>
  </GroupElement>
  <GroupElement>
    <GroupLabel level="1" reference="prodtype">A0791</GroupLabel>
    <GroupValue>34</GroupValue>
  </GroupElement>
</Grouping>
```

In future, there may be reasons (e.g. implementing use cases with 3-4 priority) expanding the grouping-structure for several nesting levels (like several fields in the GROUP BY). Currently, there are no Web services that would use this structure, but this example illustrates the idea. The example shows an arbitrary count: first by the Acquisition type and then by the Method type.

```
<Grouping name="Component Value Count By Acquisition type and Method
type" type="Level_2">
  <GroupElement>
    <GroupLabel level="1" reference="acquisitiontype">P</GroupLabel>
    <GroupLabel level="2" reference="methodtype">A</GroupLabel>
    <GroupValue>1234</GroupValue>
  </GroupElement>
  <GroupElement>
    <GroupLabel level="1" reference="acquisitiontype">P</GroupLabel>
    <GroupLabel level="2" reference="methodtype">I</GroupLabel>
    <GroupValue>456</GroupValue>
  </GroupElement>
  <GroupElement>
    <GroupLabel level="1" reference="acquisitiontype">I</GroupLabel>
    <GroupLabel level="2" reference="methodtype">A</GroupLabel>
    <GroupValue>789</GroupValue>
  </GroupElement>
  <GroupElement>
    <GroupLabel level="1" reference="acquisitiontype">I</GroupLabel>
    <GroupLabel level="2" reference="methodtype">I</GroupLabel>
    <GroupValue>1234</GroupValue>
  </GroupElement>
</Grouping>
```

## FCDB\_Describe

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

## Components

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

## Foods

Foods-element is similar to the Fod List in the FDTP.

## 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 GroupElement-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.

#### Parameters

Name	Explanation
api_userid	User application identification key. See Web services authentication
api_permission	User profile permission. See Web services authentication
fdql_sentence	See Food Data Query Language (FDQL).
version	Version of the Web service, this version 1.0 .
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.

#### Implementation

Implementation is OPTIONAL

#### 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. See Web services authentication
fdql_sentence	See Food Data Query Language (FDQL).
version	Version of the Web service, this version 1.0 .
api_signature	Message signature. See Web services authentication

### Example of the fdql\_sentence

```
<FDQL_Sentence
xsi:noNamespaceSchemaLocation="EuroFIR_Web_Service_FDQL_Sentence_versio
n_1_0.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <MetaData>
    <SchemaVersion>1.0</SchemaVersion>
    <Schema> EuroFIR_Web_Service_FDQL_Sentence_version_1_0.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 FDTP.

### Implementation

Implementation is REQUIRED

## 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. See Web services authentication
fdql_sentence	See Food Data Query Language (FDQL).
version	Version of the Web service, this version 1.0 .
api_signature	Message signature. See Web services authentication

### Example of the fdql\_sentence

```
<FDQL_Sentence
xsi:noNamespaceSchemaLocation="EuroFIR_Web_Service_FDQL_Sentence_version_1_0.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <MetaData>
    <SchemaVersion>1.0</SchemaVersion>
    <Schema> EuroFIR_Web_Service_FDQL_Sentence_version_1_0.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.

## Example

Note that StandardVocabularies and SenderInformation have been simplified in the example.

```
<EuroFIRMetaDataTransportPackage version="1.0" sentdate="2008-12-24">
  <StandardVocabularies/>
  <SenderInformation/>
  <Content datasetcreated="2008-12-24" language="tlh" acquisitiontype="F"
  domaintype="" compilationtype="">
    </Content>
    <FCDB_Describe>
      <Grouping name="FCDB Content" type="Level_1">
        <GroupElement>
          <GroupLabel level="1" reference="Food">Foods</GroupLabel>
          <GroupValue>1234</GroupValue>
        </GroupElement>
        <GroupElement>
          <GroupLabel level="1" reference="Component">Components</GroupLabel>
          <GroupValue>56</GroupValue>
        </GroupElement>
        <GroupElement>
          <GroupLabel level="1" reference="ComponentValue">Component
values</GroupLabel>
          <GroupValue>56789</GroupValue>
        </GroupElement>
      </Grouping>
    </FCDB_Describe>
    <Components>A ComponentList of available Components if
required</Components>
    <Foods>A FoodList of available Foods if required</Foods>
  </EuroFIRMetaDataTransportPackage>
```

## Implementation

Implementation is OPTIONAL

## 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. See Web services authentication
fdql_sentence	See Food Data Query Language (FDQL).
version	Version of the Web service, this version 1.0 .
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
xsi:noNamespaceSchemaLocation="EuroFIR_Web_Service_FDQL_Sentence_versio
n_1_0.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <MetaData>
    <SchemaVersion>1.0</SchemaVersion>
    <Schema> EuroFIR_Web_Service_FDQL_Sentence_version_1_0.xsd</Schema>
  </MetaData>
  <SelectClause>
    <FieldName>Count</FieldName>
  </SelectClause>
  <WhereClause>
    <Condition xsi:type="T_CommonCondition" logicalOperator="AND">
      <NameConditionField xml:lang="en">
        <FieldName>FoodName</FieldName>
      </NameConditionField>
      <ConditionOperator>LIKE</ConditionOperator>
      <ConditionValue>Avocado</ConditionValue>
    </Condition>
  </WhereClause>
</FDQL_Sentence>
```

## Semantic rules

Only allowed SELECT-clause terms is count. WHERE-clause allows only Food related terms (see reserved words). NoORDER BY-clause

## Response

Response uses the Metadata Transport Package.

## Example

Note that StandardVocabularies and SenderInformation have been simplified.

```
<EuroFIRMetaDataTransportPackage version="1.0" sentdate="2008-12-24">
  <StandardVocabularies/>
  <SenderInformation/>
  <Content datasetcreated="2008-12-24" language="tlh"
  acquisitiontype="F" domaintype="" compilationtype="">
  </Content>
  <Grouping name="Food count" type="Level_1">
    <GroupElement>
      <GroupLabel level="1" reference="Food">Foods</GroupLabel>
      <GroupValue>1234</GroupValue>
    </GroupElement>
  </Grouping>
</EuroFIRMetaDataTransportPackage>
```

## Implementation

Implementation is OPTIONAL

## 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

<b>Name</b>	<b>Explanation</b>
api_userid	User application identification key. See Web services authentication
api_permission	User profile permission. See Web services authentication
fdql_sentence	See Food Data Query Language (FDQL).
version	Version of the Web service, this version 1.0 .
api_signature	Message signature. See Web services authentication

## Example of the fdql\_sentence

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

```
<FDQL_Sentence
xsi:noNamespaceSchemaLocation="EuroFIR_Web_Service_FDQL_Sentence_version_1_0.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <MetaData>
    <SchemaVersion>1.0</SchemaVersion>
    <Schema>EuroFIR_Web_Service_FDQL_Sentence_version_1_0.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 terms is count. WHERE-clause allows only Food related terms (see reserved words). No ORDER BY-clause

## Response

Response uses the Metadata Transport Package.

## Example

Note that StandardVocabularies and SenderInformation have been simplified

```
<EuroFIRMetaDataTransportPackage version="1.0" sentdate="2008-12-24">
  <StandardVocabularies/>
  <SenderInformation/>
  <Content datasetcreated="2008-12-24" language="tlh"
acquisitiontype="F" domaintype="" compilationtype="">
  </Content>
  <Grouping name="Food Count By Product type" type="Level_1">
    <GroupElement>
      <GroupLabel level="1" reference="prodtype">A0792</GroupLabel>
      <GroupValue>12</GroupValue>
    </GroupElement>
    <GroupElement>
      <GroupLabel level="1" reference="prodtype">A0791</GroupLabel>
      <GroupValue>34</GroupValue>
    </GroupElement>
  </Grouping>
</EuroFIRMetaDataTransportPackage>
```

## Implementation

Implementation is OPTIONAL

## 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. See Web services authentication
fdql_sentence	See Food Data Query Language (FDQL).
version	Version of the Web service, this version 1.0 .
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
  xsi:noNamespaceSchemaLocation="EuroFIR_Web_Service_FDQL_Sentence_version_
  1_0.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <MetaData>
    <SchemaVersion>1.0</SchemaVersion>
    <Schema>EuroFIR_Web_Service_FDQL_Sentence_version_1_0.xsd</Schema>
  </MetaData>
  <SelectClause>
    <FieldName>FoodAllMinimum</FieldName>
    <FieldName>ComponentAllMinimum</FieldName>
    <FieldName>ComponentValueAllMinimum</FieldName>
  </SelectClause>
  <WhereClause>
    <Condition xsi:type="T_CommonCondition" logicalOperator="AND">
      <NameConditionField xml:lang="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.

## Implementation

Implementation is REQUIRED

## 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 without Components and Component Values.

### Parameters

Name	Explanation
api_userid	User application identification key. See Web services authentication
api_permission	User profile permission. See Web services authentication
fdql_sentence	See Food Data Query Language (FDQL).
version	Version of the Web service, this version 1.0 .
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 FDTP. The FDTP SHALL be marked as FoodList (see p. 29 of the FDTP)

### Implementation

Implementation is REQUIRED

### Error messages

See Error Messages and Error Codes

## GetSupportedTerms

### Description

Produces a list of terms, which the Partner Web service supports. -clauses

### Parameters

Name	Explanation
api_userid	User application identification key. See Web services authentication
api_permission	User profile permission. See Web services authentication
fdql_sentence	See Food Data Query Language (FDQL).
version	Version of the Web service, this version 1.0 .
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.

### Example

Note that StandardVocabularies, SenderInformation and Content have been simplified

```
EuroFIRMetaDataTransportPackage version="1.0" sentdate="2008-12-24">
  <StandardVocabularies/>
  <SenderInformation/>
  <Content/>
  <TermList name="Supported Where Terms">
    <Term>origfdcd</Term>
    <Term>FoodIdentifierLanguag</Term>
    <Term>FoodName</Term>
    <Term>ecompid</Term>
  </TermList>
</EuroFIRMetaDataTransportPackage>
```

**Implementation**

Implementation is OPTIONAL

**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 Web services are new services with different implementations, we do not know exactly the most probable errors. 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. The level should be selected balancing the implementation with the information needs of the user application. Currently, we are not experienced enough to give more detailed instructions but our experience will accumulate with the implementations. Meanwhile: use common sense and give the user application as much information as you would like to have if you were building it.

Table 7. Errors when the request is received.

Error message		Error code
Receive request		E1000
	Consumer errors	E1010:
		Formatting error
		Server not responding
		Unexpected response
	Server errors	E1020
		Unknown request format
		E1021

Table 8. Errors with the authentication

<b>Error message</b>			<b>Error code</b>
Authenticate Request			E2000
	Pre Authentication Request Error		E2010
		No user application identification key (api_userid	E2011
		No signature (api_signature	E2012
		No response from authentication server	E2013
	Authentication		E2020
		Invalid user application identification key (api_userid	E2021
		Invalid signature checking	E2022
		Post authentication Errors	E2030
		Access denied to user	E2031
		Undefined user application permission key (api_permission	E2032
		Parameter mismatch with the service	E2033
		Non-existing service	E2034

Table 9. Errors with the FDQL sentence processing and query

Error message		Error code
Process FDQL sentence		E3000
	Pre Processing Errors	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 group 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 group by field translation error
		E3024
		FDQL field mismatch error
		E3025
		FDQL select field mismatch error
		E3026
		FDQL where field mismatch error
		E3027
		FDQL group 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
	Data Processing	E3100
		Data source reported error - ACL Limitations
		E3101
		Data source reported error - query error
		E3102
		Data source reported error - result/viewing error
		E3103
	Post Data Process/Result Formulation	E3130
		Error in parsing dataset
		E3131

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

<b>Error message</b>			<b>Error code</b>
Return data			E4000
	Response Formulation		E4010
		Error in formulating response - missing data	E4011
		Error in XML creation	E4012
	Clean up Errors		E4020
Unknown error			E5000

## References

1. EuroFIR. *EuroFIR - Public homepage*. 2008 [cited 25 June 2008]; Available from: <http://www.eurofir.net>.
2. Becker, W., et al., *Proposal for structure and detail of a EuroFIR standard on food composition data I: Description of the standard*. 2007.
3. Becker, W., et al., *Proposal for structure and detail of a EuroFIR standard on food composition data II: Technical Annex*. 2007.
4. Møller, A. and T. Christensen, *EuroFIR Web Services - Food Data Transport Package, Version 1.3*, in *EuroFIR Technical Report*. 2008, EuroFIR.
5. LanguaL. *LanguaL Food Description Home Page*. 2008 [cited 9 December 2008]; Available from: <http://www.langual.org/>
6. LanguaL. *The LanguaL 2008 thesaurus*. 2008 [cited 9 December 2008]; Available from: <http://www.langual.org/xml/langual2008.xml>.
7. Møller, A. and J. Ireland, *LanguaL 2008. The LanguaL Thesaurus*, in *EuroFIR Technical Report*. 2008, EuroFIR.
8. Møller, A., J. Ireland, and E. Smith, *LanguaL 2008 - Introduction*, in *EuroFIR Technical Report*. 2008, EuroFIR.
9. Pakkala, H., et al., *EuroFIR Web Services: Background Report*. 2008, EuroFIR.
10. Bradner, S., *RFC 2119 Key words for use in RFCs to Indicate Requirement Levels. Best Current Practice*. 1997, Internet Engineering Task Force (IETF).
11. EuroFIR. *EuroFIR Acquisition Type Thesaurus version 1.0*. 2008 [cited 9 December 2008]; Available from: [http://www.eurofir.org/xml/EuroFIR\\_Acquisition\\_Type\\_Thesaurus\\_version\\_1.0.xml](http://www.eurofir.org/xml/EuroFIR_Acquisition_Type_Thesaurus_version_1.0.xml).
12. EuroFIR. *EuroFIR Component Thesaurus version*. 2008 [cited 9 December 2008]; Available from: [http://www.eurofir.org/xml/EuroFIR\\_Component\\_Thesaurus\\_version\\_1.0.xml](http://www.eurofir.org/xml/EuroFIR_Component_Thesaurus_version_1.0.xml).
13. EuroFIR. *EuroFIR Value Type Thesaurus version 1.0*. 2008 [cited 9 December 2008]; Available from: [http://www.eurofir.org/xml/EuroFIR\\_Value\\_Type\\_Thesaurus\\_version\\_1.0.xml](http://www.eurofir.org/xml/EuroFIR_Value_Type_Thesaurus_version_1.0.xml).
14. EuroFIR. *EuroFIR Unit Thesaurus version 1.0*. 2008 [cited 9 December 2008]; Available from: [http://www.eurofir.org/xml/EuroFIR\\_Unit\\_Thesaurus\\_version\\_1.0.xml](http://www.eurofir.org/xml/EuroFIR_Unit_Thesaurus_version_1.0.xml).
15. EuroFIR. *EuroFIR Matrix Unit Thesaurus version 1.0*. 2008 [cited 9 December 2008]; Available from: [http://www.eurofir.org/xml/EuroFIR\\_Matrix\\_Unit\\_Thesaurus\\_version\\_1.0.xml](http://www.eurofir.org/xml/EuroFIR_Matrix_Unit_Thesaurus_version_1.0.xml).
16. EuroFIR. *EuroFIR Method Type Thesaurus version 1.0*. 2008 [cited 9 December 2008]; Available from: [http://www.eurofir.org/xml/EuroFIR\\_Method\\_Type\\_Thesaurus\\_version\\_1.0.xml](http://www.eurofir.org/xml/EuroFIR_Method_Type_Thesaurus_version_1.0.xml).

17. EuroFIR. *EuroFIR Method Indicator Thesaurus version 1.0*. 2008 [cited 9 December 2008]; Available from: [http://www.eurofir.org/xml/EuroFIR\\_Method\\_Indicator\\_Thesaurus\\_version\\_1.0.xml](http://www.eurofir.org/xml/EuroFIR_Method_Indicator_Thesaurus_version_1.0.xml).
18. EuroFIR. *EuroFIR Reference Type Thesaurus version 1.0*. 2008 [cited 9 December 2008]; Available from: [http://www.eurofir.org/xml/EuroFIR\\_Reference\\_Type\\_Thesaurus\\_version\\_1.0.xml](http://www.eurofir.org/xml/EuroFIR_Reference_Type_Thesaurus_version_1.0.xml).
19. ISO, *ISO 3166-1 Codes for the representation of names of countries and their subdivisions*. 1997, International Organization for Standardization.
20. Alvestrand, H., *RFC 3066: Tags for the Identification of Languages*, in *Best Current Practice*. 2001, The Internet Society. Network Working Group.
21. Nørby, E. and A. Møller, *Prototype EuroFIR Databank – eSearch. EuroFIR WP 1.4 deliverable D1.4.4/-D1.4.5/D1.4.6*. 2007.
22. W3C. *SOAP Version 1.2 Part 1: Messaging Framework (Second Edition)*. 2007 [cited 10 December 2008]; Available from: <http://www.w3.org/TR/soap12-part1/>.
23. Yergeau, F., *RFC 3629. UTF-8, a transformation format of ISO 10646*, in *Standards track*. 2003, The Internet Society. Network Working Group.
24. Hoffman, P. and F. Yergeau, *RFC 2781. UTF-16, an encoding of ISO 10646*, in *Informational*. 2000, The Internet Society. Network Working Group.
25. W3C. *Extensible Markup Language (XML) 1.0 (Fifth Edition)*. W3C Recommendation 2008 [cited 9 December 2008]; Available from: <http://www.w3.org/TR/2008/REC-xml-20081126/>.
26. Amazon. *Amazon SimpleDB. Developer Guide*. 2008 [cited 10 December 2008]; Available from: <http://docs.amazonwebservices.com/AmazonSimpleDB/latest/DeveloperGuide/>.
27. Flickr. *Flickr Services Authentication API*. 2008 [cited 10 December 2008]; Available from: <http://www.flickr.com/services/api/auth.spec.html>.
28. Abzug, M. *MD5 Homepage (unofficial)*. 2008 [cited]; Available from: <http://userpages.umbc.edu/~mabzug1/cs/md5/md5.html>.
29. Rivest, P., *RFC 1321. The MD5 Message-Digest Algorithm*, in *Standards track*. 1992, The Internet Society. Network Working Group.

# Appendix 1 XML Schemas

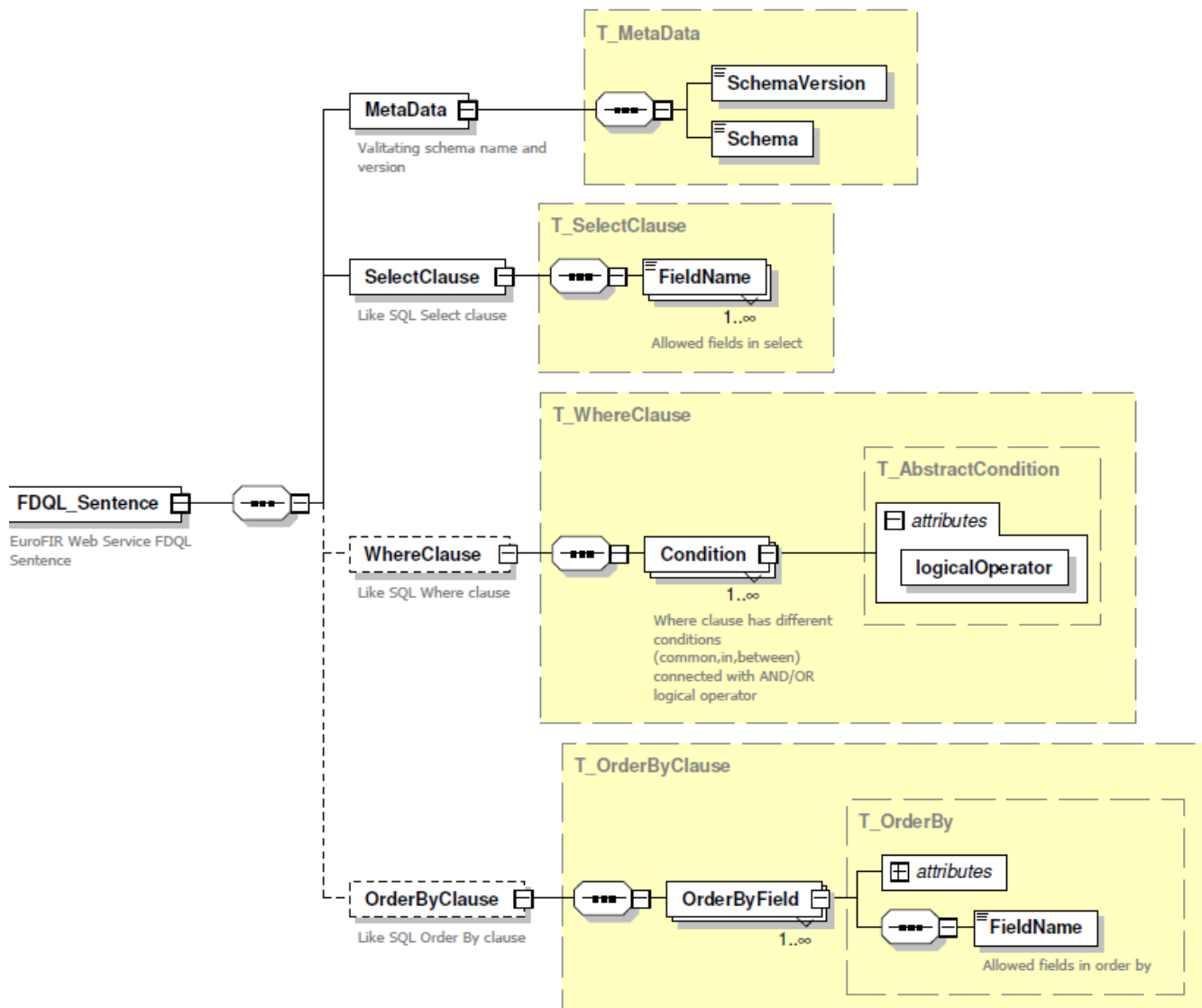
## EuroFIR\_Web\_Service\_FDQL\_Sentence\_version\_1\_0.xsd

### Documentation

Basic parts of the XML schema are presented in the Appendix 1, Figure 1. . . Metadata and SelectClause are mandatory, WhereClause and OrderByClause are optional. Each clause has a separate set of reserved words They are defined in the XML schema EuroFIR\_Web\_Service\_FDQL\_Reserved\_Words\_version\_1\_0.xsd.

The condition consists of three elements:

1. ConditionField (name of the field)
2. Operator (options limited by condition type)
3. Value

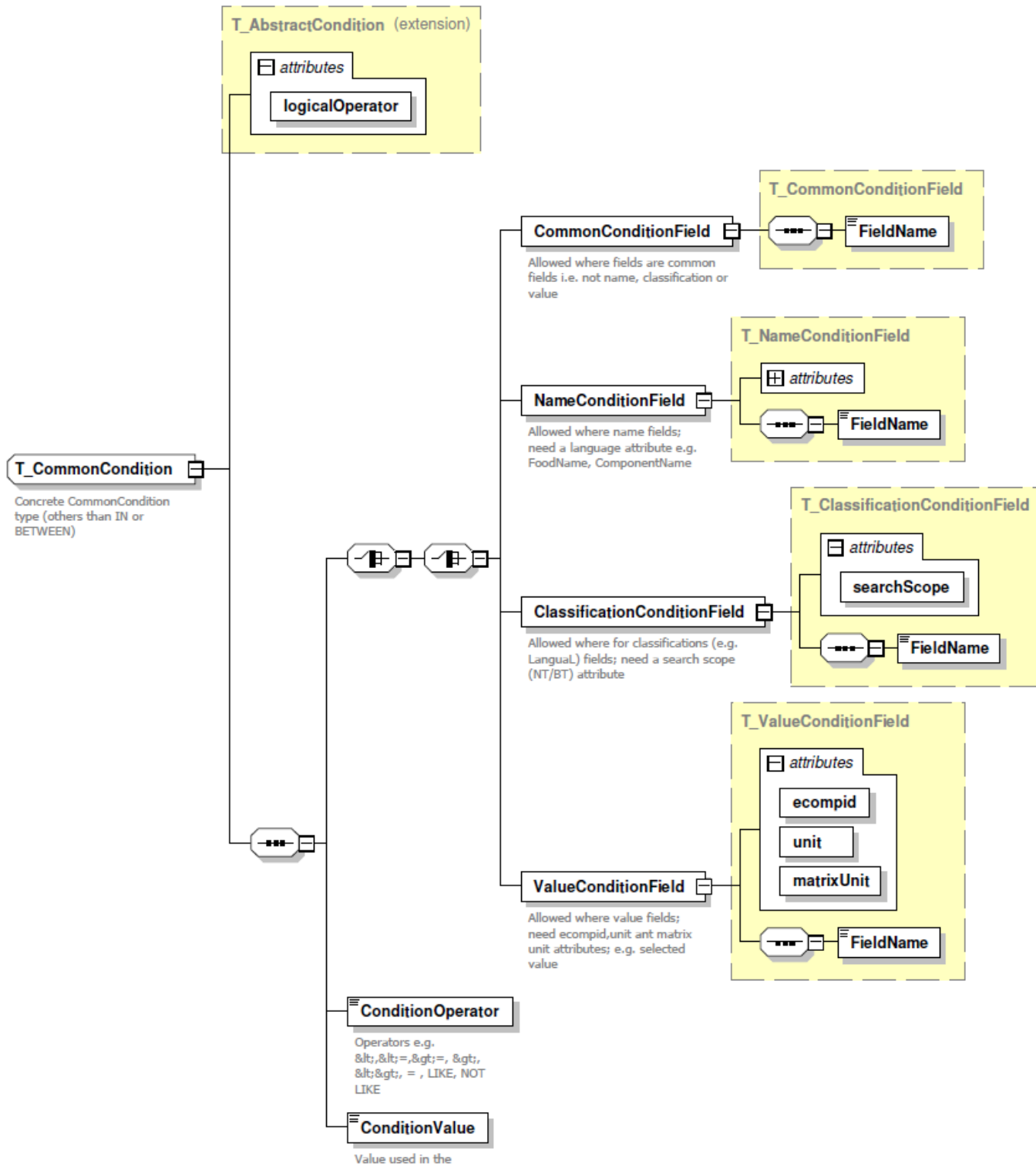


Appendix 1, Figure 1. Basic parts of the XML schema

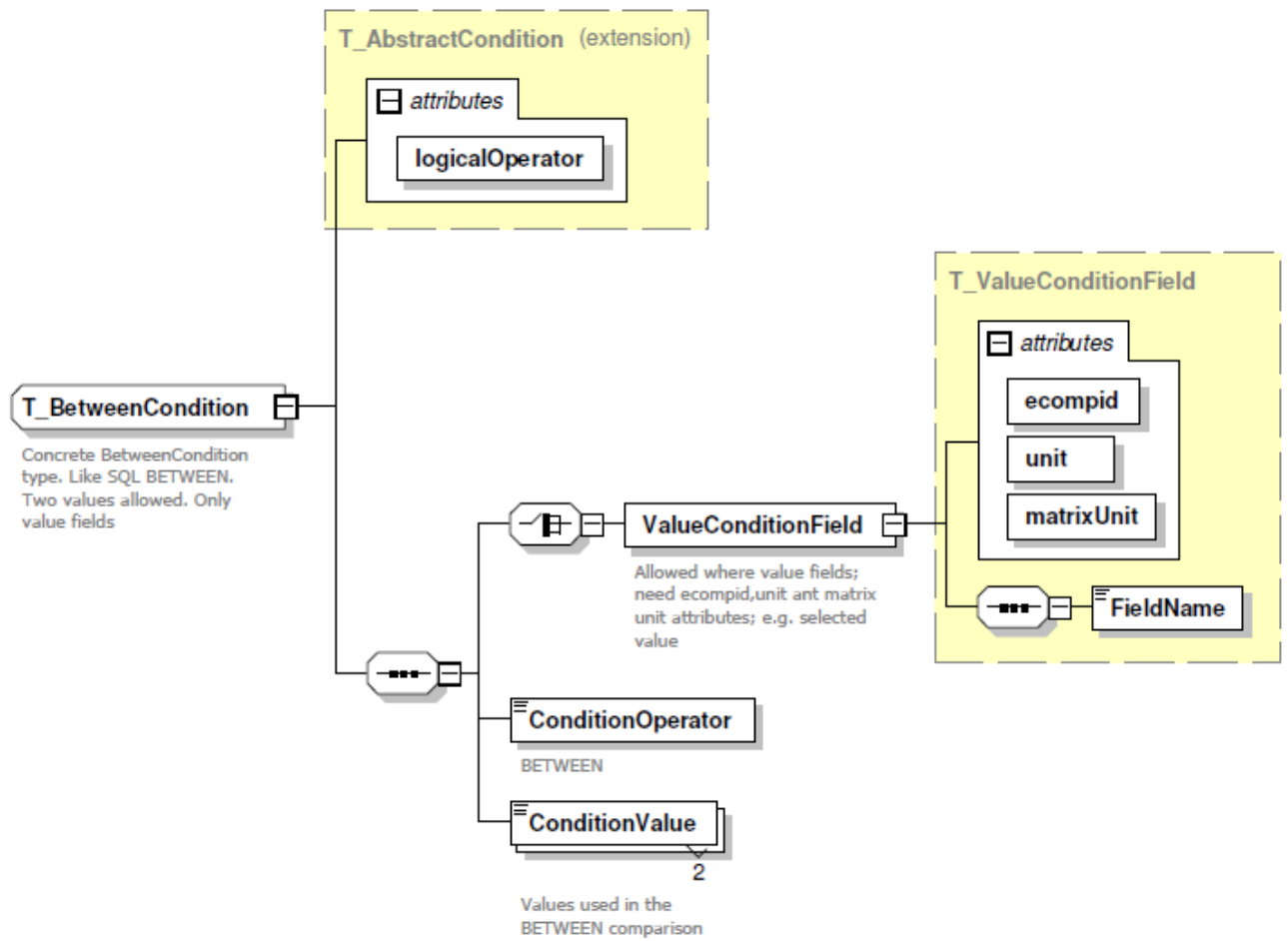
If there is more than one condition, they are linked with each other with logicalOperator –attribute. (AND/OR). These are interpreted from top to bottom (see Semantic rules).

ConditionField can be either (see Appendix 1, Figures 2-4):

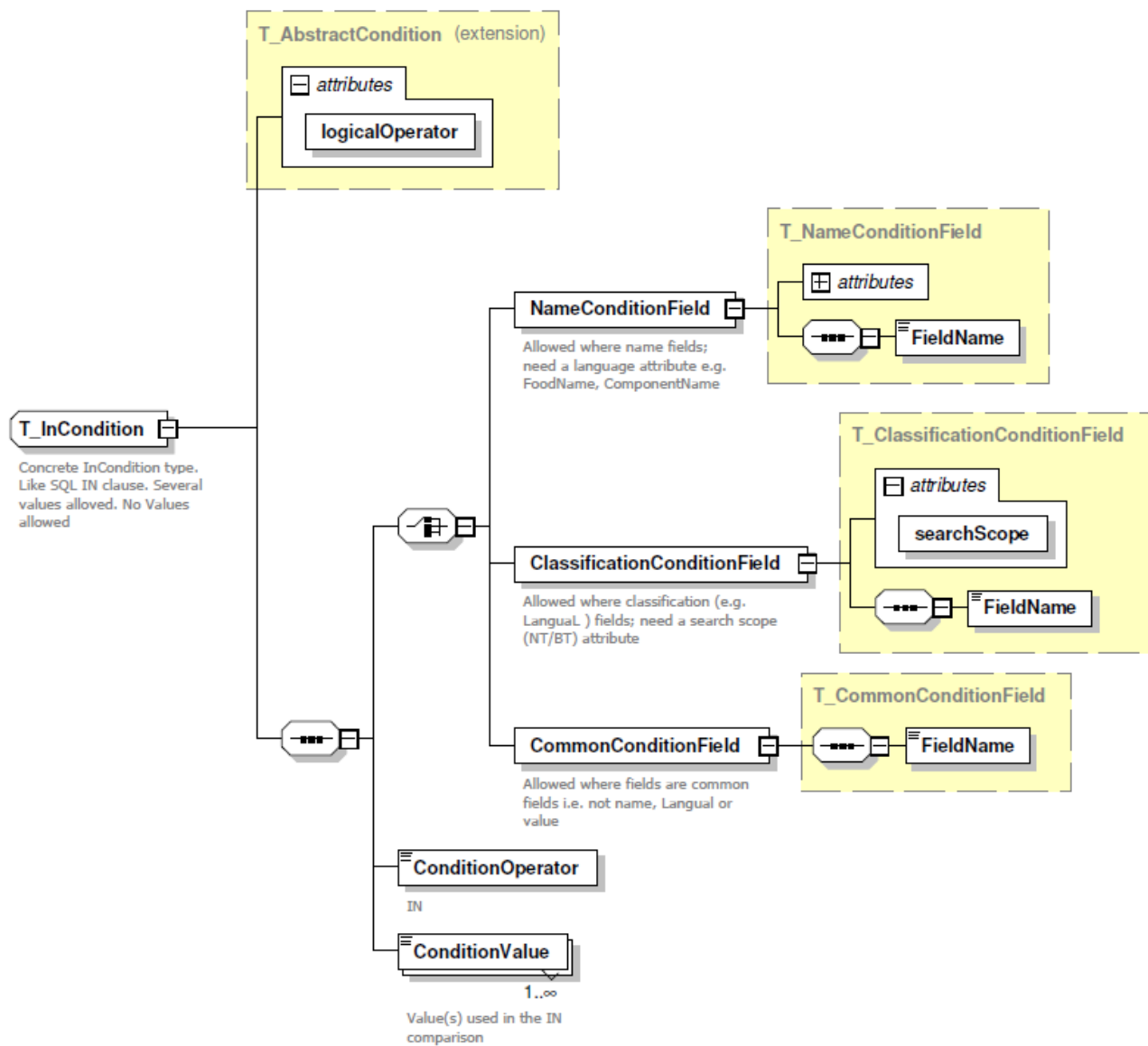
1. ClassificationConditionField. It has a mandatory attribute indicating broad (BT) or narrower term (NT) search.
2. NameConditionField. It has a mandatory attribute indicating language. Used with for food/component names.
3. ValueConditionField. It has mandatory attributes indicating ecompid, unit and matrix unit. Those vocabularies have been included in the separate schemas as enumerations. Used for the Value fields (e.g. selected value) of the Component Value.
4. CommonConditionField Other than previous. No attributes



Appendix 1, Figure 2. WhereClause Common condition: <, <=, =, <>, >, >=, LIKE, NOT LIKE (Used for other conditions than IN or BETWEEN)



Appendix 1, Figure 3. WhereClause BETWEEN condition. Only for values



Appendix 1, Figure 4. WhereClause IN condition.

## Source

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xml="http://www.w3.org/XML/1998/namespace" version="1.0">
  <xsd:include schemaLocation="EuroFIR_Unit_Thesaurus_version_1_0.xsd"/>
  <xsd:include
schemaLocation="EuroFIR_Matrix_Unit_Thesaurus_version_1_0.xsd"/>
  <xsd:include
schemaLocation="EuroFIR_Component_Thesaurus_version_1_0.xsd"/>
  <xsd:include
schemaLocation="EuroFIR_Web_Service_FDQL_Reserved_Words_version_1_0.xsd"/>
  <xsd:import namespace="http://www.w3.org/XML/1998/namespace"/>
  <!--Definitions end here-->
  <xsd:element name="FDQL_Sentence">
    <xsd:annotation>
      <xsd:documentation>EuroFIR Web Service FDQL
Sentence</xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="MetaData" type="T_MetaData">
          <xsd:annotation>
            <xsd:documentation>Valitating schema name and
version</xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="SelectClause" type="T_SelectClause">
          <xsd:annotation>
            <xsd:documentation>Like SQL Select clause</xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="WhereClause" type="T_WhereClause"
minOccurs="0">
          <xsd:annotation>
            <xsd:documentation>Like SQL Where clause</xsd:documentation>
          </xsd:annotation>
        </xsd:element>
        <xsd:element name="OrderByClause" type="T_OrderByClause"
minOccurs="0">
          <xsd:annotation>
            <xsd:documentation>Like SQL Order By
clause</xsd:documentation>
          </xsd:annotation>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <!-- XML Attribute terms -->
  <xsd:simpleType name="AT_LogicalOperator">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="AND"/>
      <xsd:enumeration value="OR"/>
      <xsd:enumeration value="NOT AND"/>
      <xsd:enumeration value="NOT OR"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="AT_searchScope">
    <xsd:annotation>
      <xsd:documentation>LanguaL or other classification search
definition Broader term (BT) or Narrower term (NT)</xsd:documentation>
    </xsd:annotation>
  </xsd:simpleType>
</xsd:schema>
```

```

    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="BT"/>
      <xsd:enumeration value="NT"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="AT_OrderingDirection">
    <xsd:annotation>
      <xsd:documentation>Like SQL ascending (ASC), descending
of the proper schema</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="ASC"/>
      <xsd:enumeration value="DESC"/>
    </xsd:restriction>
  </xsd:simpleType>
  <!-- XML Attribute terms end here -->
  <xsd:complexType name="T_MetaData">
    <xsd:annotation>
      <xsd:documentation>The Web Service may use this for the selection
of the proper schema</xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
      <xsd:element name="SchemaVersion" type="xsd:decimal"/>
      <xsd:element name="Schema" type="xsd:string"/>
    </xsd:sequence>
  </xsd:complexType>
  <xsd:complexType name="T_AbstractOperator" abstract="true">
    <xsd:annotation>
      <xsd:documentation>Abstract base comparison Operator
type</xsd:documentation>
    </xsd:annotation>
    <xsd:simpleContent>
      <xsd:extension base="xsd:string"/>
    </xsd:simpleContent>
  </xsd:complexType>
  <xsd:complexType name="T_CommonOperator">
    <xsd:annotation>
      <xsd:documentation>concrete CommonType comparison operator
(&lt;&gt;;, &gt;=, &lt;= &gt;;, &lt;;=, LIKE, NOT LIKE)</xsd:documentation>
    </xsd:annotation>
    <xsd:simpleContent>
      <xsd:restriction base="T_AbstractOperator">
        <xsd:enumeration value="="/>
        <xsd:enumeration value="&lt;&gt;"/>
        <xsd:enumeration value="&gt;;="/>
        <xsd:enumeration value="&lt;="/>
        <xsd:enumeration value="&gt;"/>
        <xsd:enumeration value="&lt;"/>
        <xsd:enumeration value="LIKE"/>
        <xsd:enumeration value="NOT LIKE"/>
      </xsd:restriction>
    </xsd:simpleContent>
  </xsd:complexType>
  <xsd:complexType name="T_BetweenOperator">
    <xsd:annotation>
      <xsd:documentation>concrete BetweenType comparison operator , like
SQL BETWEEN </xsd:documentation>
    </xsd:annotation>
    <xsd:simpleContent>
      <xsd:restriction base="T_AbstractOperator">
        <xsd:enumeration value="BETWEEN"/>
      </xsd:restriction>
    </xsd:simpleContent>
  </xsd:complexType>

```

```

        </xsd:restriction>
    </xsd:simpleContent>
</xsd:complexType>
<xsd:complexType name="T_InOperator">
    <xsd:annotation>
        <xsd:documentation>concrete InType comparison operator, like SQL
IN</xsd:documentation>
    </xsd:annotation>
    <xsd:simpleContent>
        <xsd:restriction base="T_AbstractOperator">
            <xsd:enumeration value="IN"/>
        </xsd:restriction>
    </xsd:simpleContent>
</xsd:complexType>
<xsd:complexType name="T_WhereClause">
    <xsd:sequence>
        <xsd:element name="Condition" type="T_AbstractCondition"
maxOccurs="unbounded">
            <xsd:annotation>
                <xsd:documentation>Where clause has different conditions
(common,in,between) connected with AND/OR logical
operator</xsd:documentation>
            </xsd:annotation>
        </xsd:element>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="T_SelectClause">
    <xsd:sequence>
        <xsd:element name="FieldName" type="T_SelectTerm"
maxOccurs="unbounded">
            <xsd:annotation>
                <xsd:documentation>Allowed fields in select</xsd:documentation>
            </xsd:annotation>
        </xsd:element>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="T_OrderByClause">
    <xsd:annotation>
        <xsd:documentation>Like SQL ORDER BY-clause. </xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
        <xsd:element name="OrderByField" type="T_OrderBy"
maxOccurs="unbounded"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="T_OrderBy">
    <xsd:annotation>
        <xsd:documentation>Like part of SQL ORDER BY-clause. One field with
the sorting order</xsd:documentation>
    </xsd:annotation>
    <xsd:sequence>
        <xsd:element name="FieldName" type="T_OrderByTerm">
            <xsd:annotation>
                <xsd:documentation>Allowed fields in order
by</xsd:documentation>
            </xsd:annotation>
        </xsd:element>
    </xsd:sequence>
    <xsd:attribute name="orderingDirection" type="AT_OrderingDirection"
use="required"/>
</xsd:complexType>

```

```

    <xsd:complexType name="T_CommonConditionField">
      <xsd:annotation>
        <xsd:documentation>A basic field in Where. (not Name, not
Classification, not Value)</xsd:documentation>
      </xsd:annotation>
      <xsd:sequence>
        <xsd:element name="FieldName" type="T_CommonConditionTerm"/>
      </xsd:sequence>
    </xsd:complexType>
    <xsd:complexType name="T_ClassificationConditionField">
      <xsd:annotation>
        <xsd:documentation>Field type for LanguaL or other classification.
A special case in Where.</xsd:documentation>
      </xsd:annotation>
      <xsd:sequence>
        <xsd:element name="FieldName"
type="T_ClassificationConditionTerm"/>
      </xsd:sequence>
      <xsd:attribute name="searchScope" type="AT_searchScope"
use="required"/>
    </xsd:complexType>
    <xsd:complexType name="T_ValueConditionField">
      <xsd:annotation>
        <xsd:documentation> (ComponentValue)ValueField type. A special
case in Where.</xsd:documentation>
      </xsd:annotation>
      <xsd:sequence>
        <xsd:element name="FieldName" type="T_ValueConditionTerm"/>
      </xsd:sequence>
      <xsd:attribute name="ecompid" type="AT_ecompid" use="required"/>
      <xsd:attribute name="unit" type="AT_unit" use="required"/>
      <xsd:attribute name="matrixUnit" type="AT_matrixUnit"
use="required"/>
    </xsd:complexType>
    <xsd:complexType name="T_NameConditionField">
      <xsd:annotation>
        <xsd:documentation> NameField type (either FoodName or
ComponentName) . A special case in Where.</xsd:documentation>
      </xsd:annotation>
      <xsd:sequence>
        <xsd:element name="FieldName" type="T_NameConditionTerm"/>
      </xsd:sequence>
      <xsd:attribute ref="xml:lang" use="required"/>
    </xsd:complexType>
    <xsd:complexType name="T_AbstractCondition" abstract="true">
      <xsd:annotation>
        <xsd:documentation>Abstract base Condition type</xsd:documentation>
      </xsd:annotation>
      <xsd:attribute name="logicalOperator" type="AT_LogicalOperator"
use="required"/>
    </xsd:complexType>
    <xsd:complexType name="T_InCondition">
      <xsd:annotation>
        <xsd:documentation>Concrete InCondition type. Like SQL IN clause.
Several values allowed. No Values allowed</xsd:documentation>
      </xsd:annotation>
      <xsd:complexContent>
        <xsd:extension base="T_AbstractCondition">
          <xsd:sequence>
            <xsd:choice>

```

```

        <xsd:element name="NameConditionField"
type="T_NameConditionField">
        <xsd:annotation>
        <xsd:documentation>Allowed where name fields; need a
language attribute e.g. FoodName, ComponentName</xsd:documentation>
        </xsd:annotation>
        </xsd:element>
        <xsd:element name="ClassificationConditionField"
type="T_ClassificationConditionField">
        <xsd:annotation>
        <xsd:documentation>Allowed where classification (e.g.
LanguaL ) fields; need a search scope (NT/BT)
attribute</xsd:documentation>
        </xsd:annotation>
        </xsd:element>
        <xsd:element name="CommonConditionField"
type="T_CommonConditionField">
        <xsd:annotation>
        <xsd:documentation>Allowed where fields are common fields
i.e. not name, LanguaL or value</xsd:documentation>
        </xsd:annotation>
        </xsd:element>
</xsd:choice>
<xsd:element name="ConditionOperator" type="T_InOperator">
        <xsd:annotation>
        <xsd:documentation>IN</xsd:documentation>
        </xsd:annotation>
        </xsd:element>
        <xsd:element name="ConditionValue" type="xsd:string"
maxOccurs="unbounded">
        <xsd:annotation>
        <xsd:documentation>Value(s) used in the IN
comparison</xsd:documentation>
        </xsd:annotation>
        </xsd:element>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="T_BetweenCondition">
        <xsd:annotation>
        <xsd:documentation>Concrete BetweenCondition type. Like SQL
BETWEEN. Two values allowed. Only value fields</xsd:documentation>
        </xsd:annotation>
        <xsd:complexContent>
        <xsd:extension base="T_AbstractCondition">
        <xsd:sequence>
        <xsd:choice>
        <xsd:element name="ValueConditionField"
type="T_ValueConditionField">
        <xsd:annotation>
        <xsd:documentation>Allowed where value fields; need
ecompid,unit ant matrix unit attributes; e.g. selected value
        </xsd:documentation>
        </xsd:annotation>
        </xsd:element>
</xsd:choice>
<xsd:element name="ConditionOperator" type="T_BetweenOperator">
        <xsd:annotation>
        <xsd:documentation>BETWEEN</xsd:documentation>
        </xsd:annotation>

```

```

        </xsd:element>
        <xsd:element name="ConditionValue" type="xsd:string"
minOccurs="2" maxOccurs="2">
        <xsd:annotation>
        <xsd:documentation>Values used in the BETWEEN
comparison</xsd:documentation>
        </xsd:annotation>
        </xsd:element>
    </xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="T_CommonCondition">
    <xsd:annotation>
    <xsd:documentation>Concrete CommonCondition type (others than IN or
BETWEEN)</xsd:documentation>
    </xsd:annotation>
    <xsd:complexContent>
    <xsd:extension base="T_AbstractCondition">
    <xsd:sequence>
    <xsd:choice>
    <xsd:choice>
    <xsd:element name="CommonConditionField"
type="T_CommonConditionField">
    <xsd:annotation>
    <xsd:documentation>Allowed where fields are common fields
i.e. not name, classification or value </xsd:documentation>
    </xsd:annotation>
    </xsd:element>
    <xsd:element name="NameConditionField"
type="T_NameConditionField">
    <xsd:annotation>
    <xsd:documentation>Allowed where name fields; need a
language attribute e.g. FoodName, ComponentName</xsd:documentation>
    </xsd:annotation>
    </xsd:element>
    <xsd:element name="ClassificationConditionField"
type="T_ClassificationConditionField">
    <xsd:annotation>
    <xsd:documentation>Allowed where for classifications (e.g.
Languag) fields; need a search scope (NT/BT) attribute
</xsd:documentation>
    </xsd:annotation>
    </xsd:element>
    <xsd:element name="ValueConditionField"
type="T_ValueConditionField">
    <xsd:annotation>
    <xsd:documentation>Allowed where value fields; need
ecompid,unit ant matrix unit attributes; e.g. selected
value</xsd:documentation>
    </xsd:annotation>
    </xsd:element>
    </xsd:choice>
</xsd:choice>
    <xsd:element name="ConditionOperator" type="T_CommonOperator">
    <xsd:annotation>
    <xsd:documentation>Operators e.g. <=, >, <>,
<=>, =, LIKE, NOT LIKE </xsd:documentation>
    </xsd:annotation>
    </xsd:element>
    <xsd:element name="ConditionValue" type="xsd:string">

```

```

        <xsd:annotation>
          <xsd:documentation>Value used in the
comparison</xsd:documentation>
        </xsd:annotation>
      </xsd:element>
    </xsd:sequence>
  </xsd:extension>
</xsd:complexContent>
</xsd:complexType>
</xsd:schema>

```

## EuroFIR\_Unit\_Thesaurus\_version\_1\_0.xsd

```

<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:simpleType name="AT_unit">
    <xsd:annotation>
      <xsd:documentation>All units from EuroFIR Unit Thesaurus
1.0</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="ATE"/>
      <xsd:enumeration value="BCE"/>
      <xsd:enumeration value="BX"/>
      <xsd:enumeration value="g"/>
      <xsd:enumeration value="kcal"/>
      <xsd:enumeration value="kg"/>
      <xsd:enumeration value="kJ"/>
      <xsd:enumeration value="l"/>
      <xsd:enumeration value="mg"/>
      <xsd:enumeration value="ml"/>
      <xsd:enumeration value="mmol"/>
      <xsd:enumeration value="MSE"/>
      <xsd:enumeration value="NE"/>
      <xsd:enumeration value="ng"/>
      <xsd:enumeration value="PCT"/>
      <xsd:enumeration value="R"/>
      <xsd:enumeration value="RE"/>
      <xsd:enumeration value="ug"/>
      <xsd:enumeration value="ul"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:schema>

```

## EuroFIR\_Matrix\_Unit\_Thesaurus\_version\_1\_0.xsd

```
<xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"><xsd:simpleType
name="AT_matrixUnit">
  <xsd:annotation>
    <xsd:documentation>All matrix units from EuroFIR
Matrix Unit Thesaurus 1.0</xsd:documentation>
  </xsd:annotation>
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="D"/>
    <xsd:enumeration value="DKG"/>
    <xsd:enumeration value="F"/>
    <xsd:enumeration value="FT"/>
    <xsd:enumeration value="N"/>
    <xsd:enumeration value="T"/>
    <xsd:enumeration value="TF"/>
    <xsd:enumeration value="TKG"/>
    <xsd:enumeration value="V"/>
    <xsd:enumeration value="VL"/>
    <xsd:enumeration value="VM"/>
    <xsd:enumeration value="W"/>
    <xsd:enumeration value="WKG"/>
    <xsd:enumeration value="X"/>
  </xsd:restriction>
</xsd:simpleType>
</xsd:schema>
```

## EuroFIR\_Component\_Thesaurus\_version\_1\_0.xsd

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:simpleType name="AT_ecompid">
    <xsd:annotation>
      <xsd:documentation>All EuroFIR component identifiers from
EuroFIR Component Thesaurus 1.0</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="AAA"/>
      <xsd:enumeration value="AAE-"/>
      <xsd:enumeration value="AAE10B"/>
      <xsd:enumeration value="AAS"/>
      <xsd:enumeration value="AAT-"/>
      <xsd:enumeration value="ACEAC"/>
      <xsd:enumeration value="ACESK"/>
      <xsd:enumeration value="AL"/>
      <xsd:enumeration value="ALA"/>
      <xsd:enumeration value="ALBU"/>
      <xsd:enumeration value="ALC"/>
      <xsd:enumeration value="AMMON"/>
      <xsd:enumeration value="AMYP"/>
      <xsd:enumeration value="AMYS"/>
      <xsd:enumeration value="APIGEN"/>
      <xsd:enumeration value="ARAS"/>
      <xsd:enumeration value="ARG"/>
      <xsd:enumeration value="AS"/>
      <xsd:enumeration value="ASCDL"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:schema>
```

```
<xsd:enumeration value="ASCL"/>
<xsd:enumeration value="ASH"/>
<xsd:enumeration value="ASN"/>
<xsd:enumeration value="ASF"/>
<xsd:enumeration value="ASPM"/>
<xsd:enumeration value="AVED5"/>
<xsd:enumeration value="AVED7"/>
<xsd:enumeration value="AVEDT"/>
<xsd:enumeration value="B"/>
<xsd:enumeration value="BENAC"/>
<xsd:enumeration value="BIOCHA"/>
<xsd:enumeration value="BIOT"/>
<xsd:enumeration value="BRASTR"/>
<xsd:enumeration value="BRD"/>
<xsd:enumeration value="CA"/>
<xsd:enumeration value="CADAVT"/>
<xsd:enumeration value="CAFFN"/>
<xsd:enumeration value="CAMD5"/>
<xsd:enumeration value="CAMD7"/>
<xsd:enumeration value="CAMT"/>
<xsd:enumeration value="CANTHAX"/>
<xsd:enumeration value="CAPSA"/>
<xsd:enumeration value="CAROT"/>
<xsd:enumeration value="CAROTENS"/>
<xsd:enumeration value="CARTA"/>
<xsd:enumeration value="CARTB"/>
<xsd:enumeration value="CARTBEQ"/>
<xsd:enumeration value="CARTG"/>
<xsd:enumeration value="CASN"/>
<xsd:enumeration value="CATEC"/>
<xsd:enumeration value="CD"/>
<xsd:enumeration value="CELLU"/>
<xsd:enumeration value="CHEMSC"/>
<xsd:enumeration value="CHIAC"/>
<xsd:enumeration value="CHLMP"/>
<xsd:enumeration value="CHO"/>
<xsd:enumeration value="CHOCAL"/>
<xsd:enumeration value="CHOCALOH"/>
<xsd:enumeration value="CHOLM"/>
<xsd:enumeration value="CHOLN"/>
<xsd:enumeration value="CHORL"/>
<xsd:enumeration value="CHOT"/>
<xsd:enumeration value="CHOU"/>
<xsd:enumeration value="CITAC"/>
<xsd:enumeration value="CLD"/>
<xsd:enumeration value="CO"/>
<xsd:enumeration value="CO2F"/>
<xsd:enumeration value="COLG"/>
<xsd:enumeration value="COUMEST"/>
<xsd:enumeration value="CR"/>
<xsd:enumeration value="CREATN"/>
<xsd:enumeration value="CRYPX"/>
<xsd:enumeration value="CRYPXA"/>
<xsd:enumeration value="CRYPXB"/>
<xsd:enumeration value="CU"/>
<xsd:enumeration value="CYCL"/>
<xsd:enumeration value="CYS"/>
<xsd:enumeration value="CYSTE"/>
<xsd:enumeration value="DAIDZE"/>
<xsd:enumeration value="DEN"/>
<xsd:enumeration value="DEXTN"/>
```

```
<xsd:enumeration value="DISAC"/>
<xsd:enumeration value="DOPN"/>
<xsd:enumeration value="DRYMAT"/>
<xsd:enumeration value="EDIBLE"/>
<xsd:enumeration value="ENERA"/>
<xsd:enumeration value="ENERC"/>
<xsd:enumeration value="EPICATEC"/>
<xsd:enumeration value="ERGCAL"/>
<xsd:enumeration value="ERGSTR"/>
<xsd:enumeration value="F10:0"/>
<xsd:enumeration value="F10:1"/>
<xsd:enumeration value="F10:1CIS"/>
<xsd:enumeration value="F10:1CN1"/>
<xsd:enumeration value="F10:1TRS"/>
<xsd:enumeration value="F11:0"/>
<xsd:enumeration value="F12:0"/>
<xsd:enumeration value="F12:1"/>
<xsd:enumeration value="F12:1CIS"/>
<xsd:enumeration value="F12:1CN3"/>
<xsd:enumeration value="F12:1T"/>
<xsd:enumeration value="F13:0"/>
<xsd:enumeration value="F13:0I"/>
<xsd:enumeration value="F14:0"/>
<xsd:enumeration value="F14:0AI"/>
<xsd:enumeration value="F14:0I"/>
<xsd:enumeration value="F14:1"/>
<xsd:enumeration value="F14:1CIS"/>
<xsd:enumeration value="F14:1CN5"/>
<xsd:enumeration value="F14:1TN5"/>
<xsd:enumeration value="F15+17"/>
<xsd:enumeration value="F15:0"/>
<xsd:enumeration value="F15:0AI"/>
<xsd:enumeration value="F15:0I"/>
<xsd:enumeration value="F15:1"/>
<xsd:enumeration value="F15:1CN8"/>
<xsd:enumeration value="F16:0"/>
<xsd:enumeration value="F16:0AI"/>
<xsd:enumeration value="F16:0I"/>
<xsd:enumeration value="F16:1"/>
<xsd:enumeration value="F16:1CIS"/>
<xsd:enumeration value="F16:1CN5"/>
<xsd:enumeration value="F16:1CN7"/>
<xsd:enumeration value="F16:1CN9"/>
<xsd:enumeration value="F16:1I"/>
<xsd:enumeration value="F16:1R"/>
<xsd:enumeration value="F16:1TN7"/>
<xsd:enumeration value="F16:1TRS"/>
<xsd:enumeration value="F16:2"/>
<xsd:enumeration value="F16:3"/>
<xsd:enumeration value="F16:4"/>
<xsd:enumeration value="F16:UN"/>
<xsd:enumeration value="F17:0"/>
<xsd:enumeration value="F17:0AI"/>
<xsd:enumeration value="F17:0I"/>
<xsd:enumeration value="F17:1"/>
<xsd:enumeration value="F17:1CN8"/>
<xsd:enumeration value="F18:0"/>
<xsd:enumeration value="F18:0AI"/>
<xsd:enumeration value="F18:0I"/>
<xsd:enumeration value="F18:1"/>
<xsd:enumeration value="F18:1CIS"/>
```

```
<xsd:enumeration value="F18:1CN10"/>
<xsd:enumeration value="F18:1CN11"/>
<xsd:enumeration value="F18:1CN12"/>
<xsd:enumeration value="F18:1CN13"/>
<xsd:enumeration value="F18:1CN3"/>
<xsd:enumeration value="F18:1CN4"/>
<xsd:enumeration value="F18:1CN5"/>
<xsd:enumeration value="F18:1CN6"/>
<xsd:enumeration value="F18:1CN7"/>
<xsd:enumeration value="F18:1CN8"/>
<xsd:enumeration value="F18:1CN9"/>
<xsd:enumeration value="F18:1I"/>
<xsd:enumeration value="F18:1N90"/>
<xsd:enumeration value="F18:1R"/>
<xsd:enumeration value="F18:1TN10"/>
<xsd:enumeration value="F18:1TN11"/>
<xsd:enumeration value="F18:1TN12"/>
<xsd:enumeration value="F18:1TN2"/>
<xsd:enumeration value="F18:1TN3"/>
<xsd:enumeration value="F18:1TN4"/>
<xsd:enumeration value="F18:1TN5"/>
<xsd:enumeration value="F18:1TN6"/>
<xsd:enumeration value="F18:1TN7"/>
<xsd:enumeration value="F18:1TN8"/>
<xsd:enumeration value="F18:1TN9"/>
<xsd:enumeration value="F18:1TNO"/>
<xsd:enumeration value="F18:1TRS"/>
<xsd:enumeration value="F18:2"/>
<xsd:enumeration value="F18:2CN6"/>
<xsd:enumeration value="F18:2CN9"/>
<xsd:enumeration value="F18:2CON"/>
<xsd:enumeration value="F18:2CT"/>
<xsd:enumeration value="F18:2ISO"/>
<xsd:enumeration value="F18:2R"/>
<xsd:enumeration value="F18:2TC"/>
<xsd:enumeration value="F18:2TN"/>
<xsd:enumeration value="F18:2TTN6"/>
<xsd:enumeration value="F18:3"/>
<xsd:enumeration value="F18:3CN3"/>
<xsd:enumeration value="F18:3CN6"/>
<xsd:enumeration value="F18:3N3"/>
<xsd:enumeration value="F18:3N6"/>
<xsd:enumeration value="F18:3TTN3"/>
<xsd:enumeration value="F18:4"/>
<xsd:enumeration value="F18:4CN3"/>
<xsd:enumeration value="F18:4N3"/>
<xsd:enumeration value="F19:0"/>
<xsd:enumeration value="F20:0"/>
<xsd:enumeration value="F20:0I"/>
<xsd:enumeration value="F20:1"/>
<xsd:enumeration value="F20:1CIS"/>
<xsd:enumeration value="F20:1CN11"/>
<xsd:enumeration value="F20:1CN9"/>
<xsd:enumeration value="F20:1TN11"/>
<xsd:enumeration value="F20:1TN9"/>
<xsd:enumeration value="F20:1TRS"/>
<xsd:enumeration value="F20:2"/>
<xsd:enumeration value="F20:2CN6"/>
<xsd:enumeration value="F20:2N6"/>
<xsd:enumeration value="F20:3"/>
<xsd:enumeration value="F20:3CN3"/>
```

```
<xsd:enumeration value="F20:3CN6"/>
<xsd:enumeration value="F20:3CN9"/>
<xsd:enumeration value="F20:3N3"/>
<xsd:enumeration value="F20:3N6"/>
<xsd:enumeration value="F20:4"/>
<xsd:enumeration value="F20:4CN6"/>
<xsd:enumeration value="F20:4N3"/>
<xsd:enumeration value="F20:4N6"/>
<xsd:enumeration value="F20:5"/>
<xsd:enumeration value="F20:5CN3"/>
<xsd:enumeration value="F20:5N3"/>
<xsd:enumeration value="F20:5N6"/>
<xsd:enumeration value="F21:0"/>
<xsd:enumeration value="F21:5"/>
<xsd:enumeration value="F21:5N3"/>
<xsd:enumeration value="F22:0"/>
<xsd:enumeration value="F22:1"/>
<xsd:enumeration value="F22:1CIS"/>
<xsd:enumeration value="F22:1CN11"/>
<xsd:enumeration value="F22:1CN9"/>
<xsd:enumeration value="F22:1N7"/>
<xsd:enumeration value="F22:1TN9"/>
<xsd:enumeration value="F22:2"/>
<xsd:enumeration value="F22:2CN3"/>
<xsd:enumeration value="F22:2CN6"/>
<xsd:enumeration value="F22:3CN3"/>
<xsd:enumeration value="F22:4"/>
<xsd:enumeration value="F22:4CN6"/>
<xsd:enumeration value="F22:4N3"/>
<xsd:enumeration value="F22:4N6"/>
<xsd:enumeration value="F22:5"/>
<xsd:enumeration value="F22:5CN3"/>
<xsd:enumeration value="F22:5CN6"/>
<xsd:enumeration value="F22:5N3"/>
<xsd:enumeration value="F22:5N6"/>
<xsd:enumeration value="F22:6"/>
<xsd:enumeration value="F22:6CN3"/>
<xsd:enumeration value="F22:6N3"/>
<xsd:enumeration value="F22:UN"/>
<xsd:enumeration value="F23:0"/>
<xsd:enumeration value="F24:0"/>
<xsd:enumeration value="F24:1"/>
<xsd:enumeration value="F24:1CN9"/>
<xsd:enumeration value="F24:1TN9"/>
<xsd:enumeration value="F24:2N6"/>
<xsd:enumeration value="F26:0"/>
<xsd:enumeration value="F4-10:0"/>
<xsd:enumeration value="F4-8:0"/>
<xsd:enumeration value="F4:0"/>
<xsd:enumeration value="F6:0"/>
<xsd:enumeration value="F8:0"/>
<xsd:enumeration value="FACF"/>
<xsd:enumeration value="FACID"/>
<xsd:enumeration value="FACIDCTG"/>
<xsd:enumeration value="FACN3"/>
<xsd:enumeration value="FACN6"/>
<xsd:enumeration value="FACN9"/>
<xsd:enumeration value="FAESS"/>
<xsd:enumeration value="FAFRE"/>
<xsd:enumeration value="FAMS"/>
<xsd:enumeration value="FAMSCIS"/>
```

```
<xsd:enumeration value="FAMSCXR"/>
<xsd:enumeration value="FAMSTXR"/>
<xsd:enumeration value="FAMSXR"/>
<xsd:enumeration value="FAPU"/>
<xsd:enumeration value="FAPUCR"/>
<xsd:enumeration value="FAPUCXR"/>
<xsd:enumeration value="FAPULC"/>
<xsd:enumeration value="FAPUN3"/>
<xsd:enumeration value="FAPUN3FI"/>
<xsd:enumeration value="FAPUN3VE"/>
<xsd:enumeration value="FAPUN6"/>
<xsd:enumeration value="FAPUN9"/>
<xsd:enumeration value="FAPUOT"/>
<xsd:enumeration value="FAPUTR"/>
<xsd:enumeration value="FAPUXR"/>
<xsd:enumeration value="FASAT"/>
<xsd:enumeration value="FASATR"/>
<xsd:enumeration value="FASATXR"/>
<xsd:enumeration value="FAT"/>
<xsd:enumeration value="FATAN"/>
<xsd:enumeration value="FATPL"/>
<xsd:enumeration value="FATR"/>
<xsd:enumeration value="FATUNK"/>
<xsd:enumeration value="FAUN"/>
<xsd:enumeration value="FD"/>
<xsd:enumeration value="FE"/>
<xsd:enumeration value="FIBC"/>
<xsd:enumeration value="FIBHEX"/>
<xsd:enumeration value="FIBINS"/>
<xsd:enumeration value="FIBPEN"/>
<xsd:enumeration value="FIBSOL"/>
<xsd:enumeration value="FIBT"/>
<xsd:enumeration value="FOL"/>
<xsd:enumeration value="FOLACID"/>
<xsd:enumeration value="FOLB"/>
<xsd:enumeration value="FOLFRE"/>
<xsd:enumeration value="FORMO"/>
<xsd:enumeration value="FRUS"/>
<xsd:enumeration value="FUCSTR"/>
<xsd:enumeration value="FUCSTR28"/>
<xsd:enumeration value="FUMAC"/>
<xsd:enumeration value="GALS"/>
<xsd:enumeration value="GALSD"/>
<xsd:enumeration value="GENIST"/>
<xsd:enumeration value="GLN"/>
<xsd:enumeration value="GLU"/>
<xsd:enumeration value="GLUS"/>
<xsd:enumeration value="GLUTN"/>
<xsd:enumeration value="GLY"/>
<xsd:enumeration value="GLYC"/>
<xsd:enumeration value="GLYCIT"/>
<xsd:enumeration value="GLYLIP"/>
<xsd:enumeration value="GLYRL"/>
<xsd:enumeration value="GULDKAC"/>
<xsd:enumeration value="HAEM"/>
<xsd:enumeration value="HG"/>
<xsd:enumeration value="HIS"/>
<xsd:enumeration value="HISTN"/>
<xsd:enumeration value="HYP"/>
<xsd:enumeration value="ID"/>
<xsd:enumeration value="ILE"/>
```

```
<xsd:enumeration value="INOTL"/>
<xsd:enumeration value="INULN"/>
<xsd:enumeration value="ISOCAC"/>
<xsd:enumeration value="ISOF LAVT"/>
<xsd:enumeration value="ISOMALT"/>
<xsd:enumeration value="K"/>
<xsd:enumeration value="KAEMF"/>
<xsd:enumeration value="LACAC"/>
<xsd:enumeration value="LACACD"/>
<xsd:enumeration value="LACACL"/>
<xsd:enumeration value="LACS"/>
<xsd:enumeration value="LACTTL"/>
<xsd:enumeration value="LEU"/>
<xsd:enumeration value="LIGN"/>
<xsd:enumeration value="LIGNANS"/>
<xsd:enumeration value="LUTE"/>
<xsd:enumeration value="LUTEOL"/>
<xsd:enumeration value="LUTEZEAX"/>
<xsd:enumeration value="LYCO"/>
<xsd:enumeration value="LYS"/>
<xsd:enumeration value="LYSAVL"/>
<xsd:enumeration value="MALAC"/>
<xsd:enumeration value="MALS"/>
<xsd:enumeration value="MALTRS"/>
<xsd:enumeration value="MANTL"/>
<xsd:enumeration value="MATAIRES"/>
<xsd:enumeration value="MET"/>
<xsd:enumeration value="MG"/>
<xsd:enumeration value="MK10"/>
<xsd:enumeration value="MK11"/>
<xsd:enumeration value="MK12"/>
<xsd:enumeration value="MK13"/>
<xsd:enumeration value="MK4"/>
<xsd:enumeration value="MK5"/>
<xsd:enumeration value="MK6"/>
<xsd:enumeration value="MK7"/>
<xsd:enumeration value="MK8"/>
<xsd:enumeration value="MK9"/>
<xsd:enumeration value="MN"/>
<xsd:enumeration value="MNSAC"/>
<xsd:enumeration value="MO"/>
<xsd:enumeration value="MYRIC"/>
<xsd:enumeration value="NA"/>
<xsd:enumeration value="NACL"/>
<xsd:enumeration value="NCF"/>
<xsd:enumeration value="NHAEM"/>
<xsd:enumeration value="NI"/>
<xsd:enumeration value="NIA"/>
<xsd:enumeration value="NIAAVL"/>
<xsd:enumeration value="NIAEQ"/>
<xsd:enumeration value="NIATRP"/>
<xsd:enumeration value="NITRA"/>
<xsd:enumeration value="NITRI"/>
<xsd:enumeration value="NITRN"/>
<xsd:enumeration value="NNP"/>
<xsd:enumeration value="NSP"/>
<xsd:enumeration value="NT"/>
<xsd:enumeration value="OA"/>
<xsd:enumeration value="OLSAC"/>
<xsd:enumeration value="OXALAC"/>
<xsd:enumeration value="P"/>
```

```
<xsd:enumeration value="PANTAC"/>
<xsd:enumeration value="PB"/>
<xsd:enumeration value="PECT"/>
<xsd:enumeration value="PH"/>
<xsd:enumeration value="PHE"/>
<xsd:enumeration value="PHETN"/>
<xsd:enumeration value="PHOLIP"/>
<xsd:enumeration value="PHYSTR"/>
<xsd:enumeration value="PHYTAC"/>
<xsd:enumeration value="PIPN"/>
<xsd:enumeration value="POLY"/>
<xsd:enumeration value="PORTION"/>
<xsd:enumeration value="PRO"/>
<xsd:enumeration value="PROPAC"/>
<xsd:enumeration value="PROT"/>
<xsd:enumeration value="PROTAN"/>
<xsd:enumeration value="PROTPL"/>
<xsd:enumeration value="PROTUNK"/>
<xsd:enumeration value="PSACNC"/>
<xsd:enumeration value="PSACNCI"/>
<xsd:enumeration value="PSACNCS"/>
<xsd:enumeration value="PURAC"/>
<xsd:enumeration value="PURN"/>
<xsd:enumeration value="PUTRSC"/>
<xsd:enumeration value="PYRXL"/>
<xsd:enumeration value="PYRXM"/>
<xsd:enumeration value="PYRXN"/>
<xsd:enumeration value="QUERCE"/>
<xsd:enumeration value="RAFS"/>
<xsd:enumeration value="RE"/>
<xsd:enumeration value="RETALD"/>
<xsd:enumeration value="RETOL"/>
<xsd:enumeration value="RETOL13"/>
<xsd:enumeration value="RETOLAT"/>
<xsd:enumeration value="RETOLATE"/>
<xsd:enumeration value="RETOLDH"/>
<xsd:enumeration value="RIBF"/>
<xsd:enumeration value="RIBS"/>
<xsd:enumeration value="S"/>
<xsd:enumeration value="SACCNA"/>
<xsd:enumeration value="SALAC"/>
<xsd:enumeration value="SE"/>
<xsd:enumeration value="SECORES"/>
<xsd:enumeration value="SER"/>
<xsd:enumeration value="SEROTN"/>
<xsd:enumeration value="SI"/>
<xsd:enumeration value="SITSTR"/>
<xsd:enumeration value="SOLID"/>
<xsd:enumeration value="SORAC"/>
<xsd:enumeration value="SORTL"/>
<xsd:enumeration value="SPERDN"/>
<xsd:enumeration value="SPERN"/>
<xsd:enumeration value="SPISTR"/>
<xsd:enumeration value="STARCH"/>
<xsd:enumeration value="STARES"/>
<xsd:enumeration value="STAS"/>
<xsd:enumeration value="STEROTH"/>
<xsd:enumeration value="STERT"/>
<xsd:enumeration value="STGSTR"/>
<xsd:enumeration value="STID7"/>
<xsd:enumeration value="STID7911"/>
```

```
<xsd:enumeration value="SUCAC"/>
<xsd:enumeration value="SUCS"/>
<xsd:enumeration value="SUGAD"/>
<xsd:enumeration value="SUGAN"/>
<xsd:enumeration value="SUGAR"/>
<xsd:enumeration value="TANNIN"/>
<xsd:enumeration value="TARAC"/>
<xsd:enumeration value="THEBRN"/>
<xsd:enumeration value="THIA"/>
<xsd:enumeration value="THR"/>
<xsd:enumeration value="TOCPHA"/>
<xsd:enumeration value="TOCPHB"/>
<xsd:enumeration value="TOCPHD"/>
<xsd:enumeration value="TOCPHG"/>
<xsd:enumeration value="TOCPHT"/>
<xsd:enumeration value="TOCTRA"/>
<xsd:enumeration value="TOCTRB"/>
<xsd:enumeration value="TOCTRD"/>
<xsd:enumeration value="TOCTRG"/>
<xsd:enumeration value="TOCTRI"/>
<xsd:enumeration value="TRES"/>
<xsd:enumeration value="TRP"/>
<xsd:enumeration value="TRYPN"/>
<xsd:enumeration value="TYR"/>
<xsd:enumeration value="TYRA"/>
<xsd:enumeration value="VAL"/>
<xsd:enumeration value="VITA"/>
<xsd:enumeration value="VITAACT"/>
<xsd:enumeration value="VITAPAL"/>
<xsd:enumeration value="VITB12"/>
<xsd:enumeration value="VITB6"/>
<xsd:enumeration value="VITC"/>
<xsd:enumeration value="VITD"/>
<xsd:enumeration value="VITE"/>
<xsd:enumeration value="VITK"/>
<xsd:enumeration value="VITK1"/>
<xsd:enumeration value="VITK1D"/>
<xsd:enumeration value="VITK2"/>
<xsd:enumeration value="WASTE"/>
<xsd:enumeration value="WATER"/>
<xsd:enumeration value="XYLS"/>
<xsd:enumeration value="XYLTL"/>
<xsd:enumeration value="ZEAXN"/>
<xsd:enumeration value="ZN"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:schema>
```

## EuroFIR\_Web\_Service\_FDQL\_Reserved\_Words\_version\_1\_0.xsd

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:annotation>
    <xsd:documentation>Reserved words as types</xsd:documentation>
  </xsd:annotation>
  <xsd:simpleType name="T_FoodAllTerm">
    <xsd:annotation>
      <xsd:documentation>Defines large sets of fields for
food</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="FoodAll"/>
      <xsd:enumeration value="FoodAllMandatory"/>
      <xsd:enumeration value="FoodAllMinimum"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="T_OriginalFoodClassificationTerm">
    <xsd:annotation>
      <xsd:documentation>Defines a reference to Original Food
Classification</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="origgpcd"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="T_LanguagTerm">
    <xsd:annotation>
      <xsd:documentation>Defines a reference to all Languag
thesauri</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="FoodIdentifierLanguag"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="T_FoodClassificationTerm">
    <xsd:annotation>
      <xsd:documentation>Original Food Classification or
Languag</xsd:documentation>
    </xsd:annotation>
    <xsd:union memberTypes="T_OriginalFoodClassificationTerm
T_LanguagTerm"/>
  </xsd:simpleType>
  <xsd:simpleType name="T_FoodNameTerm">
    <xsd:annotation>
      <xsd:documentation>Reference to a FoodName. The language is
defined with an attribute</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="FoodName"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="T_FoodIdTerm">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="origfdcd"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="T_FoodRecipeTerm">
```

```

    <xsd:annotation>
      <xsd:documentation>Reference to a
FoodRecipe.</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="Recipe"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="T_FoodListTerm">
    <xsd:annotation>
      <xsd:documentation>Defines a group term for food
list</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="FoodList"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="T_FoodSelectTerm">
    <xsd:annotation>
      <xsd:documentation>Food related terms allowed in the Select-
clause</xsd:documentation>
    </xsd:annotation>
    <xsd:union memberTypes="T_FoodIdTerm T_FoodNameTerm
T_FoodClassificationTerm T_FoodRecipeTerm T_FoodListTerm
T_FoodAllTerm"/>
  </xsd:simpleType>
  <xsd:simpleType name="T_FoodWhereTerm">
    <xsd:annotation>
      <xsd:documentation>Food related terms allowed in the Where-
clause</xsd:documentation>
    </xsd:annotation>
    <xsd:union memberTypes="T_FoodIdTerm T_FoodNameTerm
T_FoodClassificationTerm"/>
  </xsd:simpleType>
  <xsd:simpleType name="T_FoodOrderByTerm">
    <xsd:annotation>
      <xsd:documentation>Food related terms allowed in the OrderBy-
clause</xsd:documentation>
    </xsd:annotation>
    <xsd:union memberTypes="T_FoodIdTerm T_FoodNameTerm
T_FoodClassificationTerm"/>
  </xsd:simpleType>
  <xsd:simpleType name="T_ComponentAllTerm">
    <xsd:annotation>
      <xsd:documentation>Defines large sets of fields for
components</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="ComponentAll"/>
      <xsd:enumeration value="ComponentAllMandatory"/>
      <xsd:enumeration value="ComponentAllMinimum"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="T_ComponentNameTerm">
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="OriginalComponentName"/>
      <xsd:enumeration value="origcpnm"/>
    </xsd:restriction>
  </xsd:simpleType>
  <xsd:simpleType name="T_ComponentClassificationTerm">
    <xsd:annotation>

```

```

    <xsd:documentation>Defines a reference to EuroFIR Component
Thesaurus</xsd:documentation>
  </xsd:annotation>
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="ecompid"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="T_ComponentIdTerm">
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="origcpcd"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="T_ComponentListTerm">
  <xsd:annotation>
    <xsd:documentation>Defines a group term for component
list</xsd:documentation>
  </xsd:annotation>
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="ComponentList"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="T_ComponentSelectTerm">
  <xsd:annotation>
    <xsd:documentation>Component related terms allowed in the
Select</xsd:documentation>
  </xsd:annotation>
  <xsd:union memberTypes="T_ComponentIdTerm T_ComponentNameTerm
T_ComponentClassificationTerm T_ComponentListTerm
T_ComponentAllTerm"/>
</xsd:simpleType>
<xsd:simpleType name="T_ComponentWhereTerm">
  <xsd:annotation>
    <xsd:documentation>Component related terms allowed in the
Where</xsd:documentation>
  </xsd:annotation>
  <xsd:union memberTypes="T_ComponentIdTerm T_ComponentNameTerm
T_ComponentClassificationTerm"/>
</xsd:simpleType>
<xsd:simpleType name="T_ComponentOrderByTerm">
  <xsd:annotation>
    <xsd:documentation>Component related terms allowed in the
OrderBy</xsd:documentation>
  </xsd:annotation>
  <xsd:union memberTypes="T_ComponentIdTerm T_ComponentNameTerm
T_ComponentClassificationTerm"/>
</xsd:simpleType>
<xsd:simpleType name="T_ComponentValueAllTerm">
  <xsd:annotation>
    <xsd:documentation>Defines large sets of fields for component
value</xsd:documentation>
  </xsd:annotation>
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="ComponentValueAll"/>
    <xsd:enumeration value="ComponentValueAllMandatory"/>
    <xsd:enumeration value="ComponentValueAllMinimum"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="T_ComponentValueEntitiesTerm">
  <xsd:annotation>
    <xsd:documentation>Defines large sets of fields related to
component value entity</xsd:documentation>

```

```

</xsd:annotation>
<xsd:restriction base="xsd:string">
  <xsd:enumeration value="ComponentValue"/>
  <xsd:enumeration value="QualityIndex"/>
  <xsd:enumeration value="MethodSpecification"/>
  <xsd:enumeration value="Sample"/>
  <xsd:enumeration value="ContributingValue"/>
  <xsd:enumeration value="ValueStatistics"/>
  <xsd:enumeration value="ValueReference"/>
  <xsd:enumeration value="MethodReference"/>
  <xsd:enumeration value="NoOfAnalyticalPortionsValue"/>
</xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="T_ComponentValueCommonTerm">
  <xsd:annotation>
    <xsd:documentation>Part of the ComponentValue, fields not name,
classification or value</xsd:documentation>
  </xsd:annotation>
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="dategenerated"/>
    <xsd:enumeration value="dataevaluated"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="T_ComponentValueClassificationTerm">
  <xsd:annotation>
    <xsd:documentation>Part of the ComponentValue, includes
classifications, standard vocabularies etc</xsd:documentation>
  </xsd:annotation>
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="methodtype"/>
    <xsd:enumeration value="methodindicator"/>
    <xsd:enumeration value="valuetype"/>
    <xsd:enumeration value="acquisitiontype"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="T_ComponentValueValueTerm">
  <xsd:annotation>
    <xsd:documentation>Part of the ComponentValue, includes those
needing unit and matrixunit</xsd:documentation>
  </xsd:annotation>
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="SelectedValue"/>
    <xsd:enumeration value="Mean"/>
    <xsd:enumeration value="Median"/>
    <xsd:enumeration value="Minimum"/>
    <xsd:enumeration value="Maximum"/>
    <xsd:enumeration value="StandardDeviation"/>
    <xsd:enumeration value="StandardError"/>
    <xsd:enumeration value="StandardDeviation"/>
  </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="T_QualityIndexValueTerm">
  <xsd:annotation>
    <xsd:documentation>Field terms related to Quality index.
</xsd:documentation>
  </xsd:annotation>
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="foodidentification"/>
    <xsd:enumeration value="componentidentification"/>
    <xsd:enumeration value="samplingplan"/>
    <xsd:enumeration value="samplenumbers"/>
  </xsd:restriction>

```

```

        <xsd:enumeration value="samplehandling"/>
        <xsd:enumeration value="method"/>
        <xsd:enumeration value="performance"/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="T_NoOfAnalyticalPortionsCommonTerm">
    <xsd:annotation>
        <xsd:documentation>Part of the NoOfAnalyticalPortions, fields
not name, classification or value</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="portionsize"/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="T_NoOfAnalyticalPortionsValueTerm">
    <xsd:annotation>
        <xsd:documentation>Part of the NoOfAnalyticalPortions, includes
those needing unit and matrixunit</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="replicates"/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="T_ComponentValueSelectTerm">
    <xsd:annotation>
        <xsd:documentation>ComponentValue related terms allowed in the
Select</xsd:documentation>
    </xsd:annotation>
    <xsd:union memberTypes="T_ComponentValueAllTerm
T_ComponentValueEntitiesTerm T_ComponentValueCommonTerm
T_ComponentValueClassificationTerm T_ComponentValueValueTerm
T_NoOfAnalyticalPortionsValueTerm T_QualityIndexValueTerm
T_ComponentValueCommonTerm T_NoOfAnalyticalPortionsCommonTerm"/>
</xsd:simpleType>
<!--<xsd:simpleType name="T_ComponentValueWhereTerm">
    <xsd:annotation>
        <xsd:documentation>ComponentValue related terms allowed in the
Where. NOTE currently none</xsd:documentation>
    </xsd:annotation>
    <xsd:union memberTypes="T_ComponentValueCommonTerm
T_NoOfAnalyticalPortionsCommonTerm T_ComponentValueClassificationTerm
T_ComponentValueValueTerm T_NoOfAnalyticalPortionsValueTerm
T_QualityIndexValueTerm"/>
</xsd:simpleType-->
<xsd:simpleType name="T_ComponentValueOrderByTerm">
    <xsd:annotation>
        <xsd:documentation>ComponentValue related terms allowed in the
OrderBy</xsd:documentation>
    </xsd:annotation>
    <xsd:union memberTypes="T_ComponentValueCommonTerm
T_ComponentValueClassificationTerm T_ComponentValueValueTerm
T_QualityIndexValueTerm T_NoOfAnalyticalPortionsValueTerm"/>
</xsd:simpleType>
<xsd:simpleType name="T_MetaInformationSelectTerm">
    <xsd:annotation>
        <xsd:documentation>Defines large sets of fields related to
component value entity</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="Content"/>
        <xsd:enumeration value="FCDB Describe"/>

```

```

        <xsd:enumeration value="AvailableFoods"/>
        <xsd:enumeration value="AvailableComponents"/>
        <xsd:enumeration value="Count"/>
        <xsd:enumeration value="SupportedSelectTerms"/>
        <xsd:enumeration value="SupportedWhereTerms"/>
        <xsd:enumeration value="SupportedOrderByTerms"/>
        <!--          <xsd:enumeration value="Distribution"/>
-->
    </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="T_MetaInformationOrderByTerm">
    <xsd:annotation>
        <xsd:documentation>Defines large sets of fields related to
component value entity</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="count"/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="T_SelectTerm">
    <xsd:annotation>
        <xsd:documentation>All terms allowed in the
Select</xsd:documentation>
    </xsd:annotation>
    <xsd:union memberTypes="T_FoodSelectTerm T_ComponentSelectTerm
T_ComponentValueSelectTerm T_MetaInformationSelectTerm"/>
</xsd:simpleType>
<xsd:simpleType name="T_WhereTerm">
    <xsd:annotation>
        <xsd:documentation>Fields allowed in Where. Note: currently no
for ComponentValue</xsd:documentation>
    </xsd:annotation>
    <xsd:union memberTypes="T_FoodWhereTerm T_ComponentWhereTerm"/>
    <!--<xsd:union memberTypes="T_FoodWhereTerm T_ComponentWhereTerm
T_ComponentValueWhereTerm"/>-->
</xsd:simpleType>
<xsd:simpleType name="T_CommonConditionTerm">
    <xsd:annotation>
        <xsd:documentation>Field allowed in Where. Not LanguaL, not
Name, not Value. (currently ComponentValue not
included)</xsd:documentation>
    </xsd:annotation>
    <xsd:union memberTypes="T_FoodIdTerm"/>
    <!--          <xsd:union memberTypes="T_FoodIdTerm
T_ComponentValueCommonTerm T_NoOfAnalyticalPortionsCommonTerm"/>-->
</xsd:simpleType>
<xsd:simpleType name="T_ClassificationConditionTerm">
    <xsd:annotation>
        <xsd:documentation>Classification condition Terms Food or
Component related classification (currently ComponentValue not
included)</xsd:documentation>
    </xsd:annotation>
    <xsd:union memberTypes="T_FoodClassificationTerm
T_ComponentClassificationTerm"/>
    <!--          <xsd:union memberTypes="T_FoodClassificationTerm
T_ComponentClassificationTerm T_ComponentValueClassificationTerm"/>-->
>
</xsd:simpleType>
<xsd:simpleType name="T_NameConditionTerm">
    <xsd:annotation>

```

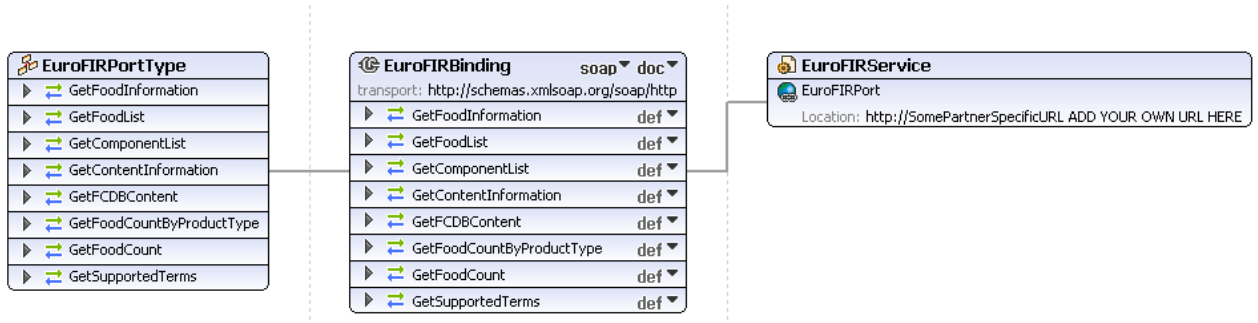
```

    <xsd:documentation>Name condition terms. Food or Component
Name</xsd:documentation>
    </xsd:annotation>
    <xsd:union memberTypes="T_FoodNameTerm T_ComponentNameTerm"/>
</xsd:simpleType>
<xsd:simpleType name="T_Empty">
    <xsd:annotation>
        <xsd:documentation>Dummy empty set</xsd:documentation>
    </xsd:annotation>
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="EmptySet_DoNotUse"/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="T_ValueConditionTerm">
    <xsd:annotation>
        <xsd:documentation>Value condition terms. Currently
none</xsd:documentation>
    </xsd:annotation>
    <xsd:union memberTypes="T_Empty"/>
    <!--      <xsd:union memberTypes="T_ComponentValueValueTerm
T_QualityIndexValueTerm T_NoOfAnalyticalPortionsValueTerm"/>-->
</xsd:simpleType>
<xsd:simpleType name="T_OrderByTerm">
    <xsd:annotation>
        <xsd:documentation>All terms allowed in the
OrderBy</xsd:documentation>
    </xsd:annotation>
    <xsd:union memberTypes="T_FoodOrderByTerm T_ComponentOrderByTerm
T_ComponentValueOrderByTerm T_MetaInformationOrderByTerm"/>
</xsd:simpleType>
</xsd:schema>

```

# Appendix 2 EuroFIR Web Service WSDL

## Documentation



Appendix 2, Figure 1. WSDL structure

### service EuroFIRService

ports	EuroFIRPort
binding	<a href="#">tns:EuroFIRBinding</a>
extensibility	<code>&lt;soap:address location="http://SomePartnerSpecificURL ADD YOUR OWN URL HERE"/&gt;</code>

binding **EuroFIRBinding**

type	tns:EuroFIRPortType
extensibility	<soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
operations	<p>GetFoodInformation  extensibility &lt;soap:operation soapAction="GetFoodInformation"/&gt;  input &lt;soap:body use="literal"/&gt;  output &lt;soap:body use="literal"/&gt;</p> <p>GetFoodList  extensibility &lt;soap:operation soapAction="GetFoodList"/&gt;  input &lt;soap:body use="literal"/&gt;  output &lt;soap:body use="literal"/&gt;</p> <p>GetComponentList  extensibility &lt;soap:operation soapAction="GetComponentList"/&gt;  input &lt;soap:body use="literal"/&gt;  output &lt;soap:body use="literal"/&gt;</p> <p>GetContentInformation  extensibility &lt;soap:operation soapAction="GetContentInformation"/&gt;  input &lt;soap:body use="literal"/&gt;  output &lt;soap:body use="literal"/&gt;</p> <p>GetFCDBCContent  extensibility &lt;soap:operation soapAction="GetFCDBCContent"/&gt;  input &lt;soap:body use="literal"/&gt;  output &lt;soap:body use="literal"/&gt;</p> <p>GetFoodCountByProductType  extensibility &lt;soap:operation  soapAction="GetFoodCountByProductType"/&gt;  input &lt;soap:body use="literal"/&gt;  output &lt;soap:body use="literal"/&gt;</p> <p>GetFoodCount  extensibility &lt;soap:operation soapAction="GetFoodCount"/&gt;  input &lt;soap:body use="literal"/&gt;  output &lt;soap:body use="literal"/&gt;</p> <p>GetSupportedTerms  extensibility &lt;soap:operation soapAction="GetSupportedTerms"/&gt;  input &lt;soap:body use="literal"/&gt;  output &lt;soap:body use="literal"/&gt;</p>
used by	Port EuroFIRPort in Service EuroFIRService

porttype **EuroFIRPortType**

operations	<p>GetFoodInformation  input tns:GetFoodInformation</p> <p>output tns:GetFoodInformationResponse</p> <p>fault tns:EuroFIRFault</p>
------------	--

	<p>GetFoodList input tns:GetFoodList</p> <p>output tns:GetFoodListResponse</p> <p>fault tns:EuroFIRFault</p>
	<p>GetComponentList input tns:GetComponentList</p> <p>output tns:GetComponentListResponse</p> <p>fault tns:EuroFIRFault</p>
	<p>GetContentInformation input tns:GetContentInformation</p> <p>output tns:GetContentInformationResponse</p> <p>fault tns:EuroFIRFault</p>
	<p>GetFCDBContent input tns:GetFCDBContent</p> <p>output tns:GetFCDBContentResponse</p> <p>fault tns:EuroFIRFault</p>
	<p>GetFoodCountByProductType input tns:GetFoodCountByProductType</p> <p>output tns:GetFoodCountByProductTypeResponse</p> <p>fault tns:EuroFIRFault</p>
	<p>GetFoodCount input tns:GetFoodCount</p> <p>output tns:GetFoodCountResponse</p> <p>fault tns:EuroFIRFault</p>
	<p>GetSupportedTerms input tns:GetSupportedTerms</p>

	output tns:GetSupportedTermsResponse  fault tns:EuroFIRFault
used by	binding EuroFIRBinding

message **GetFoodInformation**

parts	parameters  element tns:GetFoodInformation
used by	Operation GetFoodInformation in PortType EuroFIRPortType

message **GetFoodInformationResponse**

parts	parameters  element tns:EuroFIRServiceFDTPResponse
used by	Operation GetFoodInformation in PortType EuroFIRPortType

message **GetFoodList**

parts	parameters  element tns:GetFoodList
used by	Operation GetFoodList in PortType EuroFIRPortType

message **GetFoodListResponse**

parts	parameters  element tns:EuroFIRServiceFDTPResponse
used by	Operation GetFoodList in PortType EuroFIRPortType

message **GetComponentList**

parts	parameters  element tns:GetComponentList
used by	Operation GetComponentList in PortType EuroFIRPortType

message **GetComponentListResponse**

parts	parameters  element tns:EuroFIRServiceMDTPResponse
used	Operation GetComponentList in PortType EuroFIRPortType

by	
----	--

message **GetContentInformation**

parts	parameters  element tns:GetContentInformation
used by	Operation GetContentInformation in PortType EuroFIRPortType

message **GetContentInformationResponse**

parts	parameters  element tns:EuroFIRServiceFDTPResponse
used by	Operation GetContentInformation in PortType EuroFIRPortType

message **GetFCDBContent**

parts	parameters  element tns:GetFCDBContent
used by	Operation GetFCDBContent in PortType EuroFIRPortType

message **GetFCDBContentResponse**

parts	parameters  element tns:EuroFIRServiceMDTPResponse
used by	Operation GetFCDBContent in PortType EuroFIRPortType

message **GetFoodCountByProductType**

parts	parameters  element tns:GetFoodCountByProductType
used by	Operation GetFoodCountByProductType in PortType EuroFIRPortType

message **GetFoodCountByProductTypeResponse**

parts	parameters  element tns:EuroFIRServiceMDTPResponse
used by	Operation GetFoodCountByProductType in PortType EuroFIRPortType

message **GetFoodCount**

parts	parameters  element tns:GetFoodCount
-------	--

used by	Operation GetFoodCount in PortType EuroFIRPortType
---------	--

message **GetFoodCountResponse**

parts	parameters  element tns:EuroFIRServiceMDTPResponse
used by	Operation GetFoodCount in PortType EuroFIRPortType

message **GetSupportedTerms**

parts	parameters  element tns:GetSupportedTerms
used by	Operation GetSupportedTerms in PortType EuroFIRPortType

message **GetSupportedTermsResponse**

parts	parameters  element tns:EuroFIRServiceMDTPResponse
used by	Operation GetSupportedTerms in PortType EuroFIRPortType

message **EuroFIRFault**

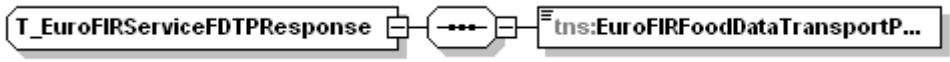
parts	parameters  element tns:EuroFIRServiceFault
used by	Operation GetFoodInformation in PortType EuroFIRPortType Operation GetFoodList in PortType EuroFIRPortType Operation GetComponentList in PortType EuroFIRPortType Operation GetContentInformation in PortType EuroFIRPortType Operation GetFCDBContent in PortType EuroFIRPortType Operation GetFoodCountByProductType in PortType EuroFIRPortType Operation GetFoodCount in PortType EuroFIRPortType Operation GetSupportedTerms in PortType EuroFIRPortType

complexType **T\_EuroFIRService**

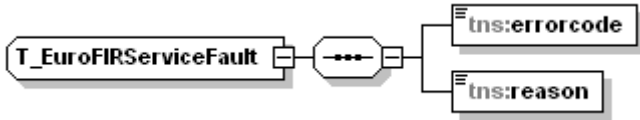
diagram	
namespace	http://eurofir.webservice.namespace
children	tns:api_userid tns:api_permission tns:fdql_sentence tns:version tns:api_signature
used by	elements GetComponentList GetContentInformation GetFCDBContent GetFoodCount

	GetFoodCountByProductType GetFoodInformation GetFoodList GetSupportedTerms
--	---


complexType **T\_EuroFIRServiceFDTPResponse**

diagram	
namespace	http://eurofir.webservice.namespace
children	<b>tns:EuroFIRFoodDataTransportPackage</b>
used by	element <b>EuroFIRServiceFDTPResponse</b>

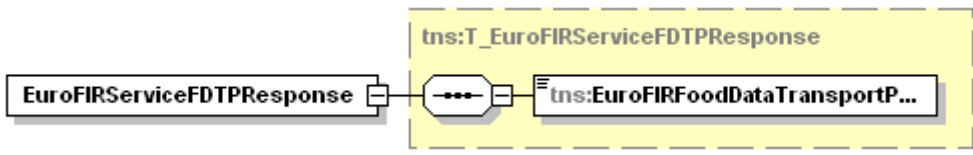
complexType **T\_EuroFIRServiceFault**

diagram	
namespace	http://eurofir.webservice.namespace
children	<b>tns:errorcode tns:reason</b>
used by	element <b>EuroFIRServiceFault</b>

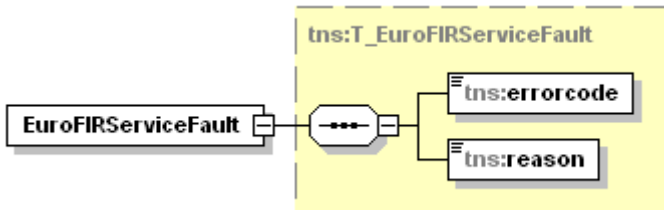
complexType **T\_EuroFIRServiceMDTPResponse**

diagram	
namespace	http://eurofir.webservice.namespace
children	<b>tns:EuroFIRMetaDataTransportPackage</b>
used by	element <b>EuroFIRServiceMDTPResponse</b>

element **EuroFIRServiceFDTPResponse**

diagram	
namespace	http://eurofir.webservice.namespace
type	<b>tns:T_EuroFIRServiceFDTPResponse</b>
properties	content complex
children	<b>tns:EuroFIRFoodDataTransportPackage</b>

element **EuroFIRServiceFault**

diagram	
namespace	http://eurofir.webservice.namespace
type	<b>tns:T_EuroFIRServiceFault</b>
properties	content complex
children	<b>tns:errorcode tns:reason</b>

element **EuroFIRServiceMDTPResponse**

diagram	
namespace	http://eurofir.webservice.namespace
type	<b>tns:T_EuroFIRServiceMDTPResponse</b>
properties	content complex
children	<b>tns:EuroFIRMetaDataTransportPackage</b>

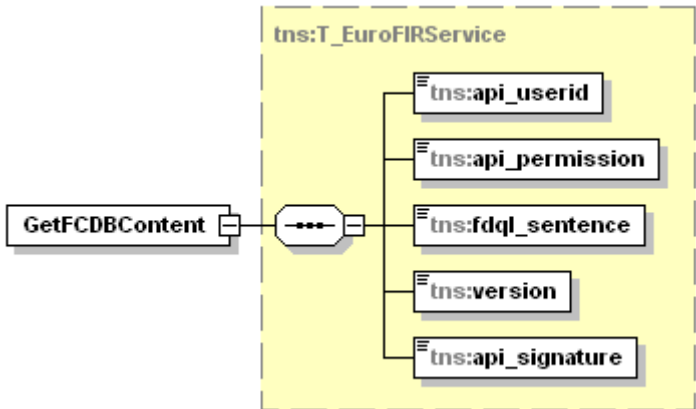
### element **GetComponentList**

diagram	
namespace	http://eurofir.webservice.namespace
type	<b>tns:T_EuroFIRService</b>
properties	content complex
children	<b>tns:api_userid tns:api_permission tns:fdql_sentence tns:version tns:api_signature</b>

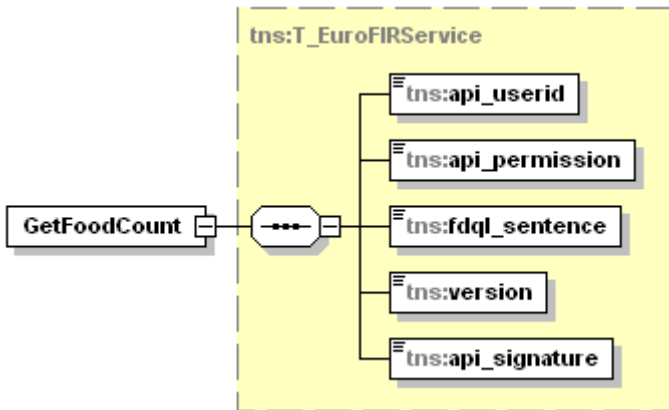
### element **GetContentInformation**

diagram	
namespace	http://eurofir.webservice.namespace
type	<b>tns:T_EuroFIRService</b>
properties	content complex
children	<b>tns:api_userid tns:api_permission tns:fdql_sentence tns:version tns:api_signature</b>

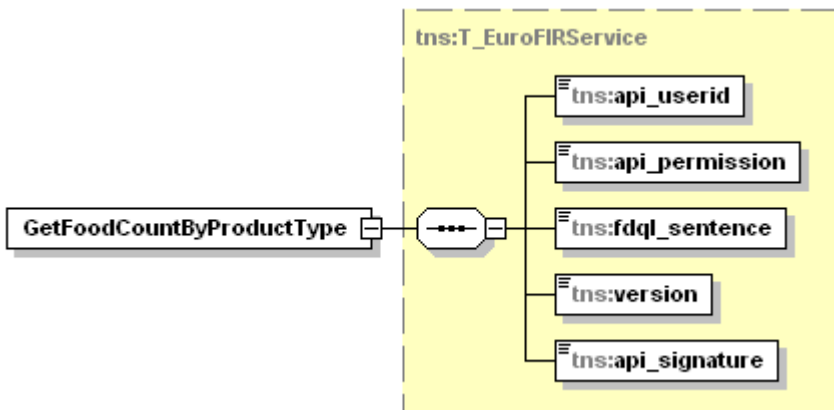
### element **GetFCDBContent**

diagram	 <p>The diagram shows the structure of the <b>GetFCDBContent</b> element. It is a complex type within the <b>tns:T_EuroFIRService</b> namespace. The element is connected to a container (represented by a circle with three dots) which contains five child elements: <b>tns:api_userid</b>, <b>tns:api_permission</b>, <b>tns:fdql_sentence</b>, <b>tns:version</b>, and <b>tns:api_signature</b>.</p>
namespace	http://eurofir.webservice.namespace
type	<b>tns:T_EuroFIRService</b>
properties	content complex
children	<b>tns:api_userid</b> <b>tns:api_permission</b> <b>tns:fdql_sentence</b> <b>tns:version</b> <b>tns:api_signature</b>

### element **GetFoodCount**

diagram	 <p>The diagram shows the structure of the <b>GetFoodCount</b> element. It is a complex type within the <b>tns:T_EuroFIRService</b> namespace. The element is connected to a container (represented by a circle with three dots) which contains five child elements: <b>tns:api_userid</b>, <b>tns:api_permission</b>, <b>tns:fdql_sentence</b>, <b>tns:version</b>, and <b>tns:api_signature</b>.</p>
namespace	http://eurofir.webservice.namespace
type	<b>tns:T_EuroFIRService</b>
properties	content complex
children	<b>tns:api_userid</b> <b>tns:api_permission</b> <b>tns:fdql_sentence</b> <b>tns:version</b> <b>tns:api_signature</b>

### element **GetFoodCountByProductType**

diagram	 <p>The diagram shows the structure of the <b>GetFoodCountByProductType</b> element. It is a complex type within the <b>tns:T_EuroFIRService</b> namespace. The element is connected to a container (represented by a circle with three dots) which contains five child elements: <b>tns:api_userid</b>, <b>tns:api_permission</b>, <b>tns:fdql_sentence</b>, <b>tns:version</b>, and <b>tns:api_signature</b>.</p>
namespace	http://eurofir.webservice.namespace
type	<b>tns:T_EuroFIRService</b>
properties	content complex
children	<b>tns:api_userid</b> <b>tns:api_permission</b> <b>tns:fdql_sentence</b> <b>tns:version</b> <b>tns:api_signature</b>

element **GetFoodInformation**

diagram	
namespace	http://eurofir.webservice.namespace
type	<b>tns:T_EuroFIRService</b>
properties	content complex
children	<b>tns:api_userid tns:api_permission tns:fdql_sentence tns:version tns:api_signature</b>

element **GetFoodList**

diagram	
namespace	http://eurofir.webservice.namespace
type	<b>tns:T_EuroFIRService</b>
properties	content complex
children	<b>tns:api_userid tns:api_permission tns:fdql_sentence tns:version tns:api_signature</b>

element **GetSupportedTerms**

diagram	
namespace	http://eurofir.webservice.namespace
type	<b>tns:T_EuroFIRService</b>
properties	content complex
children	<b>tns:api_userid tns:api_permission tns:fdql_sentence tns:version tns:api_signature</b>

## Source

```
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:tns="http://eurofir.webservice.namespace"
targetNamespace="http://eurofir.webservice.namespace">
  <wsdl:types>
    <xs:schema targetNamespace="http://eurofir.webservice.namespace"
elementFormDefault="qualified">
      <xs:complexType name="T_EuroFIRService">
        <xs:sequence>
          <xs:element name="api_userid" type="xs:token"/>
          <xs:element name="api_permission" type="xs:token"/>
          <xs:element name="fdql_sentence" type="xs:string"/>
          <xs:element name="version" type="xs:token"/>
          <xs:element name="api_signature" type="xs:token"/>
        </xs:sequence>
      </xs:complexType>
      <xs:complexType name="T_EuroFIRServiceFDTPResponse">
        <xs:sequence>
          <xs:element name="EuroFIRFoodDataTransportPackage"
type="xs:string"/>
        </xs:sequence>
      </xs:complexType>
      <xs:complexType name="T_EuroFIRServiceMDTPResponse">
        <xs:sequence>
          <xs:element name="EuroFIRMetaDataTransportPackage"
type="xs:string"/>
        </xs:sequence>
      </xs:complexType>
      <xs:complexType name="T_EuroFIRServiceFault">
        <xs:sequence>
          <xs:element name="errorcode" type="xs:token"/>
          <xs:element name="reason" type="xs:string"/>
        </xs:sequence>
      </xs:complexType>
      <xs:element name="GetFoodInformation"
type="tns:T_EuroFIRService"/>
      <xs:element name="GetFoodList" type="tns:T_EuroFIRService"/>
      <xs:element name="GetComponentList"
type="tns:T_EuroFIRService"/>
      <xs:element name="GetContentInformation"
type="tns:T_EuroFIRService"/>
      <xs:element name="GetFCDBContent" type="tns:T_EuroFIRService"/>
      <xs:element name="GetFoodCountByProductType"
type="tns:T_EuroFIRService"/>
      <xs:element name="GetFoodCount" type="tns:T_EuroFIRService"/>
      <xs:element name="GetSupportedTerms"
type="tns:T_EuroFIRService"/>
      <xs:element name="EuroFIRServiceFDTPResponse"
type="tns:T_EuroFIRServiceFDTPResponse"/>
      <xs:element name="EuroFIRServiceMDTPResponse"
type="tns:T_EuroFIRServiceMDTPResponse"/>
      <xs:element name="EuroFIRServiceFault"
type="tns:T_EuroFIRServiceFault"/>
    </xs:schema>
  </wsdl:types>
</wsdl:definitions>
```

```

</wsdl:types>
<wsdl:message name="GetFoodInformation">
  <wsdl:part name="parameters" element="tns:GetFoodInformation"/>
</wsdl:message>
<wsdl:message name="GetFoodInformationResponse">
  <wsdl:part name="parameters"
element="tns:EuroFIRServiceFDTPResponse"/>
</wsdl:message>
<wsdl:message name="GetFoodList">
  <wsdl:part name="parameters" element="tns:GetFoodList"/>
</wsdl:message>
<wsdl:message name="GetFoodListResponse">
  <wsdl:part name="parameters"
element="tns:EuroFIRServiceFDTPResponse"/>
</wsdl:message>
<wsdl:message name="GetComponentList">
  <wsdl:part name="parameters" element="tns:GetComponentList"/>
</wsdl:message>
<wsdl:message name="GetComponentListResponse">
  <wsdl:part name="parameters"
element="tns:EuroFIRServiceMDTPResponse"/>
</wsdl:message>
<wsdl:message name="GetContentInformation">
  <wsdl:part name="parameters" element="tns:GetContentInformation"/>
</wsdl:message>
<wsdl:message name="GetContentInformationResponse">
  <wsdl:part name="parameters"
element="tns:EuroFIRServiceFDTPResponse"/>
</wsdl:message>
<wsdl:message name="GetFCDBContent">
  <wsdl:part name="parameters" element="tns:GetFCDBContent"/>
</wsdl:message>
<wsdl:message name="GetFCDBContentResponse">
  <wsdl:part name="parameters"
element="tns:EuroFIRServiceMDTPResponse"/>
</wsdl:message>
<wsdl:message name="GetFoodCountByProductType">
  <wsdl:part name="parameters"
element="tns:GetFoodCountByProductType"/>
</wsdl:message>
<wsdl:message name="GetFoodCountByProductTypeResponse">
  <wsdl:part name="parameters"
element="tns:EuroFIRServiceMDTPResponse"/>
</wsdl:message>
<wsdl:message name="GetFoodCount">
  <wsdl:part name="parameters" element="tns:GetFoodCount"/>
</wsdl:message>
<wsdl:message name="GetFoodCountResponse">
  <wsdl:part name="parameters"
element="tns:EuroFIRServiceMDTPResponse"/>
</wsdl:message>
<wsdl:message name="GetSupportedTerms">
  <wsdl:part name="parameters" element="tns:GetSupportedTerms"/>
</wsdl:message>
<wsdl:message name="GetSupportedTermsResponse">
  <wsdl:part name="parameters"
element="tns:EuroFIRServiceMDTPResponse"/>
</wsdl:message>
<wsdl:message name="EuroFIRFault">
  <wsdl:part name="parameters" element="tns:EuroFIRServiceFault"/>
</wsdl:message>

```

```

<wsdl:portType name="EuroFIRPortType">
  <wsdl:operation name="GetFoodInformation">
    <wsdl:input message="tns:GetFoodInformation"/>
    <wsdl:output message="tns:GetFoodInformationResponse"/>
    <wsdl:fault name="GetFoodInformationFault"
message="tns:EuroFIRFault"/>
  </wsdl:operation>
  <wsdl:operation name="GetFoodList">
    <wsdl:input message="tns:GetFoodList"/>
    <wsdl:output message="tns:GetFoodListResponse"/>
    <wsdl:fault name="GetFoodListFault" message="tns:EuroFIRFault"/>
  </wsdl:operation>
  <wsdl:operation name="GetComponentList">
    <wsdl:input message="tns:GetComponentList"/>
    <wsdl:output message="tns:GetComponentListResponse"/>
    <wsdl:fault name="GetComponentListFault"
message="tns:EuroFIRFault"/>
  </wsdl:operation>
  <wsdl:operation name="GetContentInformation">
    <wsdl:input message="tns:GetContentInformation"/>
    <wsdl:output message="tns:GetContentInformationResponse"/>
    <wsdl:fault name="GetContentInformationFault"
message="tns:EuroFIRFault"/>
  </wsdl:operation>
  <wsdl:operation name="GetFCDBContent">
    <wsdl:input message="tns:GetFCDBContent"/>
    <wsdl:output message="tns:GetFCDBContentResponse"/>
    <wsdl:fault name="GetFCDBContentFault"
message="tns:EuroFIRFault"/>
  </wsdl:operation>
  <wsdl:operation name="GetFoodCountByProductType">
    <wsdl:input message="tns:GetFoodCountByProductType"/>
    <wsdl:output message="tns:GetFoodCountByProductTypeResponse"/>
    <wsdl:fault name="GetFoodCountByProductTypeFault"
message="tns:EuroFIRFault"/>
  </wsdl:operation>
  <wsdl:operation name="GetFoodCount">
    <wsdl:input message="tns:GetFoodCount"/>
    <wsdl:output message="tns:GetFoodCountResponse"/>
    <wsdl:fault name="GetFoodCountFault"
message="tns:EuroFIRFault"/>
  </wsdl:operation>
  <wsdl:operation name="GetSupportedTerms">
    <wsdl:input message="tns:GetSupportedTerms"/>
    <wsdl:output message="tns:GetSupportedTermsResponse"/>
    <wsdl:fault name="GetSupportedTermsFault"
message="tns:EuroFIRFault"/>
  </wsdl:operation>
</wsdl:portType>
<wsdl:binding name="EuroFIRBinding" type="tns:EuroFIRPortType">
  <soap:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
  <wsdl:operation name="GetFoodInformation">
    <soap:operation soapAction="GetFoodInformation"/>
    <wsdl:input>
      <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
      <soap:body use="literal"/>
    </wsdl:output>
    <wsdl:fault name="GetFoodInformationFault">

```

```

    <soap:fault name="GetFoodInformationFault" use="literal"/>
  </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="GetFoodList">
  <soap:operation soapAction="GetFoodList"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="GetFoodListFault">
    <soap:fault name="GetFoodListFault" use="literal"/>
  </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="GetComponentList">
  <soap:operation soapAction="GetComponentList"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="GetComponentListFault">
    <soap:fault name="GetComponentListFault" use="literal"/>
  </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="GetContentInformation">
  <soap:operation soapAction="GetContentInformation"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="GetContentInformationFault">
    <soap:fault name="GetContentInformationFault" use="literal"/>
  </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="GetFCDBContent">
  <soap:operation soapAction="GetFCDBContent"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="GetFCDBContentFault">
    <soap:fault name="GetFCDBContentFault" use="literal"/>
  </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="GetFoodCountByProductType">
  <soap:operation soapAction="GetFoodCountByProductType"/>
  <wsdl:input>
    <soap:body use="literal"/>
  </wsdl:input>
  <wsdl:output>
    <soap:body use="literal"/>
  </wsdl:output>
  <wsdl:fault name="GetFoodCountByProductTypeFault">

```

```

        <soap:fault name="GetFoodCountByProductTypeFault"
use="literal"/>
    </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="GetFoodCount">
    <soap:operation soapAction="GetFoodCount"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
    <wsdl:fault name="GetFoodCountFault">
        <soap:fault name="GetFoodCountFault" use="literal"/>
    </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="GetSupportedTerms">
    <soap:operation soapAction="GetSupportedTerms"/>
    <wsdl:input>
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output>
        <soap:body use="literal"/>
    </wsdl:output>
    <wsdl:fault name="GetSupportedTermsFault">
        <soap:fault name="GetSupportedTermsFault" use="literal"/>
    </wsdl:fault>
</wsdl:operation>
</wsdl:binding>
<wsdl:service name="EuroFIRService">
    <wsdl:port name="EuroFIRPort" binding="tns:EuroFIRBinding">
        <soap:address location="http://SomePartnerSpecificURL ADD YOUR
OWN URL HERE"/>
    </wsdl:port>
</wsdl:service>
</wsdl:definitions>

```