Open PHACTS Working Draft

Encoding units and unit types in RDF using QUDT

Open PHACTS Working Draft 13 September 2013

This version:
http://www.openphacts.org/specs/2013/WD-units-20130913/
Latest published version:
http://www.openphacts.org/specs/units/
Latest editor's draft:
http://www.bigcat.unimaas.nl/~egonw/units/
Previous version:
none
Editor:
Egon Willighagen, Maastricht University

Abstract

This guideline describes how units for data are preferably encoded, setting a standard that can then be handled uniformly by data consumers.

Status of This Document

This document is a specification by Open PHACTS. It has no official standing of any kind and does not represent the support or consensus of any standards organisation.

Disclaimer

The research leading to these results has received support from the Innovative Medicines Initiative (IMI) Joint Undertaking under grant agreement n° 115191, resources of which are composed of financial contribution from the European Union's Seventh Framework Programme (FP7/2007-2013) and EFPIA companies' in kind contribution.

Intended audience

These guidelines are intended for data providers who want to expose their data as RDF to the Open PHACTS platform.

Table of Contents

1. Introduction

This section is non-normative.

One of the goals of Open PHACTS is to provide a uniform platform for integration of data important to drug discovery [*Williams2012*]. There are many aspects of the data in the integrated resources that need consideration, but this guideline focuses on provide machine readable access to unit and unit type information.

The importance of making that information machine readable is that it allow consumers to search for data in a uniform way. That is, the platform no longer relies on data providers to use the same units. For data providers it means that they can just provide their data in the original units. For example, the ChEMBL database tracks both the units used in the data sources (e.g. literature) as well as report a standard unit. That allows the ChEMBL web interface to search drugs with IC_{50} value below a certain molarity expressed in nM. However, a major downside of this approach is that it requires to standardize these values explicitly in the database, and it requires search interfaces to still use this unit.

2. Document Conventions

2.1 RDF Syntax and Namespaces

All examples in this document are written in the Turtle RDF syntax [*TURTLE*]. Throughout the document, the following namespaces are used:

```
@prefix chembl: <http://rdf.farmbio.uu.se/chembl/onto/#> .
@prefix qudt:   <http://qudt.org/1.1/schema/qudt#> .
@prefix rdf:    <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs:   <http://www.w3.org/2000/01/rdf-schema#> .
@prefix dc:     <http://purl.org/dc/elements/1.1/> .
@prefix owl:    <http://www.w3.org/2002/07/owl#> .
@prefix qudt:   <http://qudt.org/schema/qudt#> .
@prefix skos:   <http://www.w3.org/2004/02/skos/core#> .
@prefix xsd:    <http://www.w3.org/2001/XMLSchema#> .
@prefix ops:    <http://www.openphacts.org/units/> .
@prefix uo:     <http://purl.obolibrary.org/obo/> .
```

3. The QUDT Ontology

This section is non-normative.

A solution to this is to use a unit ontology, like the QUDT ontology (Quantities, Units, Dimensions, and Types). Using such semantic annotation, we can reason on how to convert units of the same type into each other. For example, it allows us to convert micromolar into nanomolar automatically. Even more, if search engines use the ontology too, the user can search in any molar unit type, and the data can capture the data in any unit type.

In fact, the QUDT ontology does not only provide entries for units, it also provides a full framework to express how units can be interconverted. Of course, this is limited to units of the same type, e.g. molar concentration. It does not provide us means to convert a molar concentration into a weight per volume unit. That would require additional information on the subject of study, such as the molecular weight, but also information about purity.

3.1 Libraries: jQUDT for Java

Ontologies are easier to use when there are bindings for programming languages. For Java the jQUDT library has been developed [*Willighagen2012jQUDT*]. This library provides a functionality to deal with quantities expressed in QUDT units. For example, we can create a Unit object, reflecting a particular unit:

```
Unit unit1 = new Unit();
unit1.setResource(new URI("http://qudt.org/vocab/unit#Kelvin"));
```

Alternatively, you can use a factory:

```
UnitFactory factory = UnitFactory.getInstance();
Unit unit = factory.getUnit("http://qudt.org/vocab/unit#Kelvin");
```

The factory has additional methods that, for example, allow you to retrieve all temperature units, taking advantage of the unit classification in the QUDT ontology:

```
UnitFactory factory = UnitFactory.getInstance();
List units = factory.getURIs("http://qudt.org/schema/qudt#TemperatureUnit");
```

4. Encoding measurements

This section is non-normative.

If we have a measure quantity, we can associate the value and the unit of that quantity. We can use custom predicated for that, as in this example from ChEMBL-RDF:

```
chembl:a31876 chembl:standardValue "24000.0"^^xsd:float .
chembl:a31876 chembl:standardUnit <http://www.openphacts.org/units/Nanomolar> .
```

We explictly encoded the literal to be a float, using the XML Schema data type xsd:float.

4.1 Example: ChEMBL-RDF

The ChEMBL-RDF data sets (prior to ChEMBL 15) are created using a set of PHP and Groovy scripts [*Willighagen2013*]. The input of these scripts, a relational database, provides textual representations of units. Therefore, the matching between ChEMBL data and QUDT units has to be explicitly made. The script that converts the activities is written in Groovy, an extension of Java, and a Java Map is used, instantiated in the Groovy syntax:

Example

```
unitMappings = [
  nM:"http://www.openphacts.org/units/Nanomolar",
  uM:"http://www.openphacts.org/units/Micromolar",
  mM:"http://www.openphacts.org/units/Millimolar",
  pM:"http://www.openphacts.org/units/Picomolar",
  "%":"http://qudt.org/schema/qudt#floatPercentage",
  "ug.mL-1":"http://www.openphacts.org/units/MicrogramPerMilliliter",
  "ug/ml":"http://www.openphacts.org/units/MicrogramPerMilliliter",
  "ug ml-1":"http://www.openphacts.org/units/MicrogramPerMilliliter",
  "pg ml-1":"http://www.openphacts.org/units/PicogramPerMilliliter",
]
```

Using this information, we can look up which units we are exposing as QUDT, and create triples accordingly. The ChEMBL-RDF scripts use the following Groovy code:

Example

```
// this returns the "standard_units" field of the SQL query result row
units = row.standard_units
if (units != null && unitMappings.containsKey(units)) {
  // the Java map as given earlier
  qudtUnits = unitMappings.get(units)

  // first output the value as QUDT unit as RDF using a OpenRDF repository connection "con"
  con.add(actURI,
    factory.createURI(OPS + "standardValue"),
```

```
      factory.createLiteral((float)row.standard_value)
    )
    con.add(actURI,
      factory.createURI(OPS + "standardUnit"),
      factory.createURI(qudtUnits)
    )
  }
```

This code results in RDF as given earlier:

Example

```
chembl:a31876 chembl:standardValue "24000.0"^^xsd:float .
chembl:a31876 chembl:standardUnit <http://www.openphacts.org/units/Nanomolar> .
```

5. Encoding units

This section is non-normative.

The QUDT ontology comes with an extensive list of units, but not all are defined. However, the ontology provides the means to specify further units using the ontology itself. For example, for Open PHACTS various additional units have been defined.

In the following example, we will define the millimolar unit. For this, we first need to define the molar. A molar is of the MolarConcentrationUnit type and a SI-derived unit. We further specify human readable label, abbreviation, and symbol. We also link to DBPedia for further information. But, importantly, we define the information we need to convert units of the same type into each other. For this we define an offset and a multiplier. Only the latter is needed for conversion from nanomolar to millimolar (etc), but the offset we may need for other unit types, such as temperature when you like to convert Kelvin into Celsius. For molar we then get the following base type:

```
ops:Molar
 rdf:type qudt:MolarConcentrationUnit , qudt:SIDerivedUnit ;
 rdfs:label "Molar"^^xsd:string ;
 qudt:abbreviation "M"^^xsd:string ;
 qudt:conversionMultiplier
  1000 ;
 qudt:conversionOffset
  "0.0"^^xsd:double ;
 qudt:symbol "mol/dm^3"^^xsd:string ;
 skos:exactMatch <http://dbpedia.org/resource/Molar_concentration> .
```

We note that we have a multiplier from the most basic concentration unit in QUDT, which (unfortunately) is not explicitly referenced in this definition.

For variants we basically just repeat this definition, but then with different conversion coefficients. For example, for millimolar we can define:

```
ops:Millimolar
 rdf:type qudt:MolarConcentrationUnit , qudt:SIDerivedUnit ;
 rdfs:label "Millimolar"^^xsd:string ;
 qudt:abbreviation "mM"^^xsd:string ;
 qudt:conversionMultiplier
  1 ;
 qudt:conversionOffset
  "0.0"^^xsd:double ;
 qudt:symbol "mmol/dm^3"^^xsd:string .
```

6. Converting units

Because the ontology defines how units of the same base type can be interconverted, this can be computationally performed. For example, we can use the jqudt library for this [*Willighagen2012jQUDT*], as shown in the below Java source code example.

Source:

```
Quantity obs = new Quantity(0.1, Micromolar.getInstance());
System.out.println(obs + " = " +  obs.convertTo(Nanomolar.getInstance()));
```

```
Quantity temp = new Quantity(20, TemperatureUnit.CELSIUS);
System.out.println(temp + " = " +  temp.convertTo(TemperatureUnit.KELVIN));
```

Output:

```
0.1 µM = 100.00000000000001 nM
20.0 C = 293.0 K
```

Unit Ontology support

jQUDT knows about some mappings between resources from the Unit Ontology (UO) [*UO2013*] and QUDT, allowing (some) units to be constructed from UO URIs too.

Example

```
UnitOntologyFactory factory = UnitOntologyFactory.getInstance();
Unit unit = factory.getUnit("http://purl.obolibrary.org/obo/UO_0000065");
```

A full list of mappings can be found in Appendix B.

A. Units defined by Open PHACTS extending QUDT

The following list is the list of additional units by Open PHACTS. The list is currently available from the Open PHACTS GitHub repository.

```
@prefix dc:      <http://purl.org/dc/elements/1.1/> .
@prefix owl:     <http://www.w3.org/2002/07/owl#> .
@prefix qudt:    <http://qudt.org/schema/qudt#> .
@prefix rdf:     <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs:    <http://www.w3.org/2000/01/rdf-schema#> .
@prefix skos:    <http://www.w3.org/2004/02/skos/core#> .
@prefix xsd:     <http://www.w3.org/2001/XMLSchema#> .
@prefix ops:     <http://www.openphacts.org/units/> .

ops:Molar
 rdf:type qudt:MolarConcentrationUnit , qudt:SIDerivedUnit ;
 rdfs:label "Molar"^^xsd:string ;
 qudt:abbreviation "M"^^xsd:string ;
 qudt:conversionMultiplier
  1000 ;
 qudt:conversionOffset
  "0.0"^^xsd:double ;
 qudt:symbol "mol/dm^3"^^xsd:string ;
 skos:exactMatch <http://dbpedia.org/resource/Molar_concentration> .

ops:Millimolar
 rdf:type qudt:MolarConcentrationUnit , qudt:SIDerivedUnit ;
 rdfs:label "Millimolar"^^xsd:string ;
 qudt:abbreviation "mM"^^xsd:string ;
 qudt:conversionMultiplier
  1 ;
 qudt:conversionOffset
  "0.0"^^xsd:double ;
 qudt:symbol "mmol/dm^3"^^xsd:string .

ops:Micromolar
 rdf:type qudt:MolarConcentrationUnit , qudt:SIDerivedUnit ;
 rdfs:label "Micromolar"^^xsd:string ;
 qudt:abbreviation "µM"^^xsd:string ;
 qudt:conversionMultiplier
  0.001 ;
 qudt:conversionOffset
  "0.0"^^xsd:double ;
 qudt:symbol "µmol/dm^3"^^xsd:string .

ops:Nanomolar
 rdf:type qudt:MolarConcentrationUnit , qudt:SIDerivedUnit ;
 rdfs:label "Nanomolar"^^xsd:string ;
 qudt:abbreviation "nM"^^xsd:string ;
 qudt:conversionMultiplier
  0.000001 ;
```

```
 qudt:conversionOffset
  "0.0"^^xsd:double ;
 qudt:symbol "nmol/dm^3"^^xsd:string .

ops:Picomolar
 rdf:type qudt:MolarConcentrationUnit , qudt:SIDerivedUnit ;
 rdfs:label "Picomolar"^^xsd:string ;
 qudt:abbreviation "pM"^^xsd:string ;
 qudt:conversionMultiplier
  0.000000001 ;
 qudt:conversionOffset
  "0.0"^^xsd:double ;
 qudt:symbol "pmol/dm^3"^^xsd:string .

ops:GramPerLiter
 rdf:type qudt:MassPerVolumeUnit, qudt:SIDerivedUnit, qudt:DerivedUnit ;
 rdfs:label "Gram per Liter"^^xsd:string ;
 qudt:abbreviation "g/L"^^xsd:string ;
 qudt:conversionMultiplier 1.0 ;
 qudt:conversionOffset 0.0 ;
 qudt:symbol "g/dm^3"^^xsd:string .

ops:MicrogramPerMilliliter
 rdf:type qudt:MassPerVolumeUnit, qudt:SIDerivedUnit, qudt:DerivedUnit ;
 rdfs:label "Microgram per Milliliter"^^xsd:string ;
 qudt:abbreviation "µg/mL"^^xsd:string ;
 qudt:conversionMultiplier 1.0 ;
 qudt:conversionOffset 0.0 ;
 qudt:symbol "µg/cm^3"^^xsd:string .

ops:PicogramPerMilliliter
 rdf:type qudt:MassPerVolumeUnit, qudt:SIDerivedUnit, qudt:DerivedUnit ;
 rdfs:label "Picogram per Milliliter"^^xsd:string ;
 qudt:abbreviation "pg/mL"^^xsd:string ;
 qudt:conversionMultiplier 0.000001 ;
 qudt:conversionOffset 0.0 ;
 qudt:symbol "pg/cm^3"^^xsd:string .

ops:MilligramPerDeciliter
 rdf:type qudt:MassPerVolumeUnit, qudt:SIDerivedUnit, qudt:DerivedUnit ;
 rdfs:label "Milligram per Deciliter"^^xsd:string ;
 qudt:abbreviation "mg/dL"^^xsd:string ;
 qudt:conversionMultiplier 100.0 ;
 qudt:conversionOffset 0.0 .

ops:Nanometer
 rdf:type qudt:LengthUnit, qudt:SIDerivedUnit, qudt:DerivedUnit ;
 rdfs:label "Nanometer"^^xsd:string ;
 qudt:abbreviation "nm"^^xsd:string ;
 qudt:conversionMultiplier 0.00000001 ;
 qudt:conversionOffset 0.0 .

ops:SquareAngstrom
 rdf:type qudt:AreaUnit, qudt:SIDerivedUnit ;
 rdfs:label "Square Ångström"^^xsd:string ;
 qudt:abbreviation "Å^2"^^xsd:string ;
 qudt:conversionMultiplier
  0.00000000000000000001 ;
 qudt:conversionOffset
  "0.0"^^xsd:double ;
 qudt:symbol "Å^2"^^xsd:string .

ops:Milligram
 rdf:type qudt:MassUnit, qudt:SIDerivedUnit ;
 rdfs:label "Milligram"^^xsd:string ;
 qudt:abbreviation "mg"^^xsd:string ;
 qudt:conversionMultiplier
  0.000001 ;
 qudt:conversionOffset
  "0.0"^^xsd:double ;
 qudt:symbol "mg"^^xsd:string .

ops:Microgram
 rdf:type qudt:MassUnit, qudt:SIDerivedUnit ;
 rdfs:label "Microgram"^^xsd:string ;
 qudt:abbreviation "µg"^^xsd:string ;
 qudt:conversionMultiplier
  0.000000001 ;
 qudt:conversionOffset
```

```
  "0.0"^^xsd:double ;
 qudt:symbol "µg"^^xsd:string .

ops:Nanogram
 rdf:type qudt:MassUnit, qudt:SIDerivedUnit ;
 rdfs:label "Nanogram"^^xsd:string ;
 qudt:abbreviation "ng"^^xsd:string ;
 qudt:conversionMultiplier
  0.000000000001 ;
 qudt:conversionOffset
  "0.0"^^xsd:double ;
 qudt:symbol "µg"^^xsd:string .

ops:Picogram
 rdf:type qudt:MassUnit, qudt:SIDerivedUnit ;
 rdfs:label "Picogram"^^xsd:string ;
 qudt:abbreviation "pg"^^xsd:string ;
 qudt:conversionMultiplier
  0.000000000000001 ;
 qudt:conversionOffset
  "0.0"^^xsd:double ;
 qudt:symbol "pg"^^xsd:string .

ops:Femtogram
 rdf:type qudt:MassUnit, qudt:SIDerivedUnit ;
 rdfs:label "Femtogram"^^xsd:string ;
 qudt:abbreviation "fg"^^xsd:string ;
 qudt:conversionMultiplier
  0.000000000000000001 ;
 qudt:conversionOffset
  "0.0"^^xsd:double ;
 qudt:symbol "fg"^^xsd:string .

ops:MilligramPerKilogram
 rdf:type qudt:SIDerivedUnit ;
 rdfs:label "Milligram per Kilogram"^^xsd:string ;
 qudt:abbreviation "mg/kg"^^xsd:string ;
 qudt:symbol "mg/kg"^^xsd:string .

ops:Millimole
 rdf:type qudt:AmountOfSubstanceUnit, qudt:SIDerivedUnit ;
 rdfs:label "Millimole"^^xsd:string ;
 qudt:abbreviation "mmol"^^xsd:string ;
 qudt:conversionMultiplier
  0.001 ;
 qudt:conversionOffset
  "0.0"^^xsd:double ;
 qudt:symbol "mmol"^^xsd:string .

ops:Micromole
 rdf:type qudt:AmountOfSubstanceUnit, qudt:SIDerivedUnit ;
 rdfs:label "Micromole"^^xsd:string ;
 qudt:abbreviation "µmol"^^xsd:string ;
 qudt:conversionMultiplier
  0.000001 ;
 qudt:conversionOffset
  "0.0"^^xsd:double ;
 qudt:symbol "µmol"^^xsd:string .

ops:Nanomole
 rdf:type qudt:AmountOfSubstanceUnit, qudt:SIDerivedUnit ;
 rdfs:label "Nanomole"^^xsd:string ;
 qudt:abbreviation "nmol"^^xsd:string ;
 qudt:conversionMultiplier
  0.000000001 ;
 qudt:conversionOffset
  "0.0"^^xsd:double ;
 qudt:symbol "nmol"^^xsd:string .

ops:Picomole
 rdf:type qudt:AmountOfSubstanceUnit, qudt:SIDerivedUnit ;
 rdfs:label "Picomole"^^xsd:string ;
 qudt:abbreviation "pmol"^^xsd:string ;
 qudt:conversionMultiplier
  0.000000000001 ;
 qudt:conversionOffset
  "0.0"^^xsd:double ;
 qudt:symbol "pmol"^^xsd:string .
```

```
ops:Femtomole
 rdf:type qudt:AmountOfSubstanceUnit, qudt:SIDerivedUnit ;
 rdfs:label "Femtomole"^^xsd:string ;
 qudt:abbreviation "fmol"^^xsd:string ;
 qudt:conversionMultiplier
  0.000000000000001 ;
 qudt:conversionOffset
  "0.0"^^xsd:double ;
 qudt:symbol "fmol"^^xsd:string .

ops:Milliliter
 rdf:type qudt:VolumeUnit, qudt:SIDerivedUnit ;
 rdfs:label "Milliliter"^^xsd:string ;
 qudt:abbreviation "mL"^^xsd:string ;
 qudt:conversionMultiplier
  0.001 ;
 qudt:conversionOffset
  "0.0"^^xsd:double ;
 qudt:symbol "mL"^^xsd:string .

ops:Microliter
 rdf:type qudt:VolumeUnit, qudt:SIDerivedUnit ;
 rdfs:label "Microliter"^^xsd:string ;
 qudt:abbreviation "µL"^^xsd:string ;
 qudt:conversionMultiplier
  0.000001 ;
 qudt:conversionOffset
  "0.0"^^xsd:double ;
 qudt:symbol "µL"^^xsd:string .

ops:PartsPerMillion
 rdf:type qudt:QuantityKind ;
 qudt:quantityKind quantity:DimensionlessRatio ;
 rdfs:label "Parts per Million"^^xsd:string ;
 qudt:symbol "ppm"^^xsd:string .
```

B. Mappings between the Unit Ontology and QUDT

This table excludes the following Unit Ontology units for which no mapping has been created yet: uo:UO_0000197, uo:UO_0000198, uo:UO_0000271, uo:UO_0000272, and uo:UO_0000311.

Unit Ontology	QUDT
uo:EFO_0004374	ops:MilligramPerDeciliter
uo:EFO_0004385	ops:PicogramPerMilliliter
uo:UO_0000009	qudt:Kilogram
uo:UO_0000010	qudt:SecondTime
uo:UO_0000015	qudt:Centimeter
uo:UO_0000016	qudt:Millimeter
uo:UO_0000017	qudt:Micrometer
uo:UO_0000018	ops:Nanometer
uo:UO_0000021	qudt:Gram
uo:UO_0000022	ops:Milligram
uo:UO_0000023	ops:Microgram
uo:UO_0000024	ops:Nanogram
uo:UO_0000025	ops:Picogram
uo:UO_0000026	ops:Femtogram
uo:UO_0000027	qudt:DegreeCelsius
uo:UO_0000028	qudt:Millisecond
uo:UO_0000031	qudt:MinuteTime
uo:UO_0000032	qudt:Hour
uo:UO_0000033	qudt:Day
uo:UO_0000039	qudt:Micromole
uo:UO_0000040	qudt:Millimole
uo:UO_0000041	qudt:Nanomole
uo:UO_0000042	qudt:Picomole

uo:UO_0000043	qudt:Femtomole
uo:UO_0000062	ops:Molar
uo:UO_0000063	ops:Millimolar
uo:UO_0000064	ops:Micromolar
uo:UO_0000065	ops:Nanomolar
uo:UO_0000066	ops:Picomolar
uo:UO_0000073	ops:Femtomolar
uo:UO_0000098	ops:Milliliter
uo:UO_0000099	qudt:Liter
uo:UO_0000101	ops:Microliter
uo:UO_0000169	ops:PartsPerMillion
uo:UO_0000173	ops:GramPerMilliliter
uo:UO_0000175	ops:GramPerLiter
uo:UO_0000176	ops:MilligramPerMilliliter
uo:UO_0000187	qudt:Percent
uo:UO_0000274	ops:MicrogramPerMilliliter
uo:UO_0000275	ops:NanogramPerMilliliter
uo:UO_0000308	ops:MilligramPerKilogram

C. References

C.1 Normative references

No normative references.

C.2 Informative references

[TURTLE]
D. Beckett, T. Berners-Lee, E. Prud'hommeaux. Turtle - Terse RDF Triple Language, W3C Working Draft 09 August 2011. http://www.w3.org/TR/turtle/

[UO2013]
Unit Ontology (2013). http://code.google.com/p/unit-ontology/.

[Williams2012]
Williams, A. J., Harland, L., Groth, P., Pettifer, S., Chichester, C., Willighagen, E. L., Evelo, C. T., et al. (2012). Open PHACTS: Semantic interoperability for drug discovery. Drug Discovery Today. http://dx.doi.org/10.1016/j.drudis.2012.05.016.

[Willighagen2012jQUDT]
Willighagen, E.L. (2012), http://github.com/egonw/jqudt/

[Willighagen2013]
Willighagen, E.L., Waagmeester, A., Spjuth, O., Ansell, P., Williams, A.J., Tkachenko, V., Hastings, J., Chen, B., Wild, D.J. (2013). The ChEMBL database as Linked Open Data, J. Cheminformatics. http://dx.doi.org/.