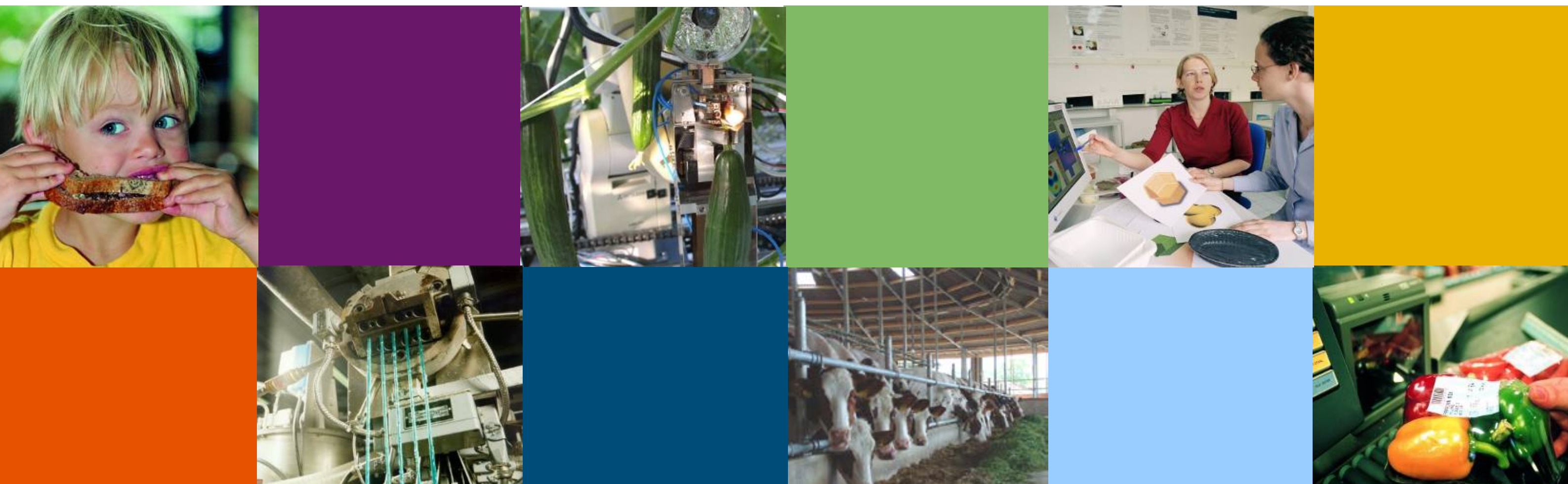
OM 2.0

Deliverable Q2 2015

eFoodlab WP2

Hajo Rijgersberg
Mari Wigham
Don Willems
Jan Top

Report 1596



Colophon

Title	OM 2.0
Author(s)	H. Rijgersberg, M. Wigham, D. Willems, J. Top
Number	1596
ISBN-number	978-94-6257-526-4
Date of publication	30-06-2015
Confidentiality	No
Approved by	Jan Top

Wageningen UR Food & Biobased Research
P.O. Box 17
NL-6700 AA Wageningen
Tel: +31 (0)317 480 084
E-mail: info.fbr@wur.nl
Internet: www.wur.nl

1 Introduction

OM, the ontology of units of measure and related concepts, is an ontology that is used across the world. The ontology defines, apart from units, such as metre and kilogram, also quantities, such as length and mass, and other attributes that are important for expressing quantitative values. The idea is that digital data are better accessible and understandable for computer tools using ontologies, by expressing them in ontologies. A units ontology is essential for expressing quantitative data. Formalization and semantics prevent problems of misinterpretation, both by human and computer. As a result, more can be done with existing data, preferably in an automated way. Errors or even disasters due to different units of measure can occur in many common-place activities, such as the transfer of designs from R&D to production, research institutes in an international collaboration project, or cooperation between different companies on the same construction project. An intriguing example of the latter is the Mars orbiter that was lost because units were omitted in data and different units were assumed, causing a loss of $125 million[1].

The ontology is about ten years old now. It is not the only ontology in the world that describes the given domain. QUDT (Quantities, Units, Dimensions and Types; Masters et al., 2010) of NASA and UO (Unit Ontology; Gkoutos et al., 2012) of OBO Foundry are at the moment the most well-known. However, these ontologies, and also the older ontologies that have become obsolete in the meantime, have shortcomings. QUDT does not, for example, have prefixes, like milli, kilo, etc. These are of primary importance in the domain of units, for scaling and relating units to each other according to standard rules. UO does contain prefixes, but lacks other important items, such as for example quantities. Contrary to all other ontologies, OM is grounded in an informal description drafted from official sources of instances like the National Institute of Standards and Technology (NIST)[2], integrating the different sources, on basis of which the formal description of the ontology was subsequently constructed. The entire exercise is described in (Rijgersberg, 2013). OM and accompanying web services for, for example, conversion of units can be used, downloaded, and distributed freely from Wurvoc.org.

Based on requests from users as well as our own use of the ontology in data integration tasks, we have now developed a new version of the ontology: OM 2.0, in which we make the ontology more broadly applicable and increase the quality of the definitions. We describe the adaptations and the underlying rationale in the next section. In Section 3 we conclude.

[1] *CNN Tech, September 30, 1999.*
[2] *www.nist.gov*

2 Problems and adaptations

From a user's perspective, there were a number of problems regarding the previous version of OM, number 1.8. The most urgent problem was that users sometimes found that the quantities or units that they needed were not in OM, or the quality of their definitions was not high enough. We have tackled these problems by making the following adaptations, which, along with more detailed descriptions of the problems, are described in the subsections below:

1. Definition of special units with numbers, such as l/100 km
2. Extending the definitions of quantities
3. Removing redundancy
4. Removing rarely-used quantities
5. Improving readability
6. Making OM easier to maintain

In order to be able to follow the explanation below, we first briefly introduce some technical computer terms that will be used in the text. Generally, we will discuss *objects*, i.e., computer representations of things from reality, for example the object "cat". An object has *properties*, for example "hasNumberOfPaws". Properties have *ranges*, these are the types of the values that properties have. For example, "hasNumberOfPaws" is typically of the type (or range) integer, which is the set of natural numbers (0, 1, 2, etc.). Ranges can also relate to object types. For example, the range of the property 'hasKitten' would be of the type "cat". For computers it is necessary to state all this explicitly in order to enable automated data processing of data.

2.1 Definition of special units with numbers, such as l/100 km

We regularly received requests from users to define units with numerical prefixes that deviate from the ones existing in OM. One eye-catching example is the widely used unit for fuel consumption, l/100 km. The denominator of this compound unit[3], 100 km, cannot be represented by an existing prefix (such as centi, kilo, micro, etc.) that would represent the factor 100 000. Such a prefix does, quite simply, not exist. Therefore we have created, apart from the class PrefixedUnit which accommodates combinations of singular units (such as metre, gram, and second) with prefixes (kilo, nano, mega, etc.), a class UnitMultiple, which can be used to link any unit, such as kilometre, to a factor, such as 100. In this way the unit multiple 100 km can be defined, which can be used in the unit l/100 km.
In order to enable non-standard units to occur in compound units, we have now extended the range of the operands to include not only units, but also measures. In this way, the unit in question, l/100 km, can be constructed from the unit litre and the measure 100 km.

[3] *Compound units are units that are composed from two or more other units, such as m/s, m^2, and kg/m^3.*

2.2 Extending the definitions of quantities

2.2.1 Context property for binary quantities

Quantities in OM have a type and a phenomenon. The hasPhenomenon property indicates the object where the quantity is related to. The quantity 'lengthOfMyTable', for example, is of the type Length, and has a hasPhenomenon property with the value myTable, a concept from an external ontology.

The previous version of OM only contained one such hasPhenomenon property. The environment or the surroundings, or the *context* of the quantity is not expressed. However, quantities frequently exist that involve *two* phenomena, such as in the concentration of sugar in water, the mass fraction of an ingredient in a sample, etc. These kinds of quantities are so-called *binary* quantities.

The user could arrange this by defining compound objects (objects that contain other objects). However, this would not express the role of the different objects explicitly. To facilitate the idea of binary quantities, we have now defined a property to complement hasPhenomenon, i.e., the hasContext property. The first property, the hasPhenomenon property, remains as it is, it is not renamed.

2.2.2 Aggregate function property for quantities

In datasets, one often encounters quantities with additional annotation, such as the maximum voltage v_{max}, the average mass of an animal m_{av}, etc. The values of these quantities arise from specific aggregation functions, such as maximum and average. In standard query languages, such as SQL and SPARQL, these are called *aggregate functions*. We use the same terminology in OM 2.0. We have defined the property "hasAggregateFunction" of quantities, with range "Function", a new class which has the instances first, last, min, max, sum, avg, count, stddev, and prod. It is likely that this list will be extended in the future. Using this property, now aggregate quantities can be defined, such as 'theAverageMassOfMyCat', which is a quantity of the type Mass, hasPhenomenon 'cat' and hasAggregateFunction 'avg'.

2.2.3 Compound quantities

Just like compound *units* (e.g. m/s, m^2, kg/m^3), compound *quantities* are used in practice, such as area per volume, force length, etc. In order to facilitate this, we have defined the class CompoundQuantity (subclass of Quantity), with subclasses QuantityMultiplication, QuantityDivision, and QuantityExponentiation. This is the same structure as was already defined for CompoundUnit (a subclass of UnitOfMeasure): UnitMultiplication, UnitDivision, and UnitExponentiation. The ranges of the original properties of the compound units, hasTerm1, hasTerm2 (for multiplications), hasNumerator and hasDenominator (for divisions), and hasBase (for exponentiations) are extended with quantities. Subsequently we have restricted the ranges to units of measure for compound units, and to quantities for compound quantities. This way a compound quantity such as force per area can be defined, as a quantity division with hasNumerator Force and hasDenominator Area.

2.3 Removing redundancy

In the previous version of OM, the class 'measure' had the properties numerical_value and unit_of_measure_or_measurement_scale. Using the latter property one could refer to not only units of measure, but also measurement scales, such as for expressing absolute temperatures. An example of the first type of measure is 3 m, which refers to a unit, metre, and the value 3. However, 4 K can be defined in two ways: with the unit kelvin, which generally indicates a temperature difference, or with the Kelvin scale, indicating an absolute temperature of −269.15 °C. However, in OM, measurement scales already have points for this purpose. One particular use of points on a scale is that of defining so-called fixed points (a subclass of Point in OM). A measurement scale is spanned by fixed points, such as the boiling point of water for defining the point 373.15 K on the Kelvin scale. As the concept for points on a scale was already available, it is redundant to use a combination of a numerical value and measurement scale to for example refer to an absolute temperature. Therefore we have replaced the unit_of_measure_or_measurement_scale property with a hasUnitOfMeasure property, which has as its range only units of measure. Whereas before "3 °C" (a measure) could either refer to a unit (degree Celsius) or a measurement scale (the Celsius scale), now it can only refer to the unit (degree Celsius). However, quantities now may have instances of the class Measure as well as Point as their value. In this way, for instance, a quantity "theTemperatureOfMyDog" can have as its value a point on a temperature scale, e.g., "40OnTheCelsiusScale". The latter indicates an absolute temperature. This is different from the value '40 °C', because then it is undetermined whether a temperature difference (of 40 K) or an absolute temperature (of 313.15 K) is expressed. This modification in OM removes an ambiguity in the use of measures.

2.4 Removing rarely-used quantities

In OM a large amount of "isolated" quantities were present – quantities that were not related to units. 'Degree_of_ionization_for_charge_number_z_greater_than_or_equal_to_1' is one such example. Mostly these concerned stub quantities, concepts with an empty or inadequate definition. To improve the quality of definitions in OM, we have removed the most exotic quantities, only relevant for a very specific domain, now. For the particular domains for which these quantities are relevant, we will have to find out which quantities and units are used. This can have to be done by literature research and interviewing experts.

2.5 Improving readability

The previous version of OM used underscores to separate words in concept identifiers. Now we use PascalCase for classes, e.g. UnitOfMeasure rather than Unit_of_measure, and camelCase for properties and instances. PacalCase and camelCase are widely regarded as easily readable formats. In addition we now use verbs to express properties, following the standard approach in the Semantic Web.

2.6 Making OM easier to maintain

Previously, OM was stored in one large file, which we have now divided into several files, each one of them comprising a specific domain. This makes the ontology easier to maintain. Moreover, the exercise has led to an important improvement of the domains (called application areas in OM) that are defined in the ontology. A number of application areas have been added, merged or removed.

3 Conclusion

Given practical experience with OM over the past years, we have made a number of changes to OM, resulting in OM 2.0. The most significant adaptations are the use of measures in compound units, the aggregate function property for quantities, and the context property for binary quantities. As a result, more high-level support in quantitative data and computations can be developed. The ontology can be downloaded from Wurvoc.org, and freely used and distributed, just like the previous version of OM. OM 2.0 is not backwards compatible with the previous version of the ontology, OM 1.8. Therefore OM 1.8 will remain available for data using this ontology.

We will extend OM further to include more quantities and units. For example, we are currently defining astronomical quantities and units by analyzing which quantities and units occur in scientific papers in this domain. There are plans to integrate appropriate units and quantities from QUDT and UO. Finally, we intend to include physical constants with their values. All open issues relating to the structure of OM have been addressed in this latest release. As such we regard OM 2.0 as stable, and do not plan any changes to the structure of the ontology in the near future.

Finally, we hope to propose OM as a W3C standard.

References

Gkoutos, G.V., Schofield, P.N., Hoehndorf, R. The Units Ontology: a tool for integrating units of measurement in science. Database 2012: bas033. http://database.oxfordjournals.org/content/2012/bas033.abstract.

Masters, J., Hodgson, R., Keller, P.J. QUDT – Quantities, Units, Dimensions and Data Types in OWL and XML, 2010, http://www.qudt.org.

Rijgersberg, H., Semantic support for quantitative research. Ph.D. thesis, 2013.

W3C. JSON-LD 1.0. A JSON-based Serialization for Linked Data. W3C Recommendation 16 January 2014. The World Wide Web Consortium (W3C), 2014.