Project no. **004074**

Project acronym: **NATURNET-REDIME**

Project title: **New Education and Decision Support Model for Active Behaviour in Sustainable Development Based on Innovative Web Services and Qualitative Reasoning**

Instrument: **SPECIFIC TARGETED RESEARCH PROJECT**

Thematic Priority: **SUSTDEV-2004-3.VIII.2.e**

D2.3.1 Document Type Definition (DTD) for QR Model Fragments[1] Redime

Due date of deliverable: **01/03/2006**
Actual submission date: **24/03/2006**

Start date of project: **1st March 2005** Duration: **30 months**

Organisation name of lead contractor for this deliverable:
University of Amsterdam

Revision: Final

Project co-funded by the European Commission within the Sixth Framework Programme (2002-2006)		
Dissemination Level		
PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

[1] AUTHORS: Jochem Liem & Bert Bredeweg

Abstract

The Naturnet-Redime project needs to share qualitative knowledge models of issues relevant to sustainable development. To be able to store, share and reuse models and model fragments within a community building qualitative models, a format needs to be defined that allows the exchange of these fragments via the internet. This document describes the formalisation of qualitative models and model fragments in the Web Ontology Language (OWL).

Document history

Version	Status	Date	Author
1.0	Final	24/03/2006	Liem, J. & Bredeweg, B.

Contents

1 Introduction

Qualitative Reasoning and Modelling (QRM) is an area of research within Artificial Intelligence (AI) that deals with conceptual knowledge. It is an innovative technique that involves non-numerical descriptions of systems and their behaviour, preserving all the important behavioural properties and distinctions.

The Naturnet-Redime project needs to share qualitative knowledge models of issues relevant to sustainable development. The envisioned solution to this issue is creating a qualitative model repository which can be indexed using an ontology. To be able to truly search for specific structures in qualitative models using such a repository, these models have to be represented using a formalism which is open and accessible. The default file format of the qualitative reasoning and modelling tool Garp3 is a binary file, which is unsuitable for this purpose.

The desire to share and reuse knowledge has led to the establishment of the Web Ontology Language (OWL) [1] knowledge representation language. OWL seems the obvious choice for representing and sharing qualitative models. The different ontologies distinguished by Heist et al. [9] inspired the different ontologies distinguished for the qualitative reasoning domain (see figure 1). The representation ontology is the Web Ontology Language. The qualitative vocabulary is represented in the generic ontology. Qualitative models are formalised as domain ontologies and refer to the generic ontology.

This technical document describes how the qualitative vocabulary and the qualitative reasoning models are formalised in OWL. Chapter 2 describes the Web Ontology Language. Chapter 3 explains how the well-formedness and consistency of the ontology was checked, the notation used in the rest of the document, and the ModelIngredient class. The rest of the chapters describe the Structural Building Blocks, Behavioural Building Blocks, Behavioural Dependencies, Aggregates, Special Model Ingredients. The final chapter discussed the results and some future work.

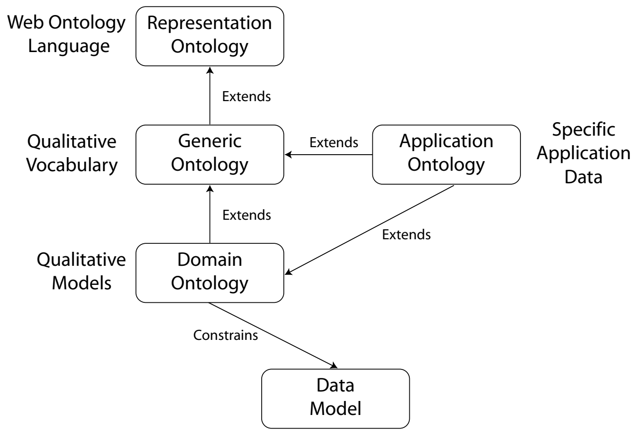

Figure 1: The QR ingredient taxonomy defining the QR vocabulary.

2 The Web Ontology Language

For the purpose of this project, the Web Ontology Language (OWL) [19, 1, 17] was chosen as the knowledge representation language to formalise qualitative reasoning vocabulary and models in. To be able to ascertain and describe the problems associated with the formalisation of AI ontologies, this section describes the KR constructs available in OWL.

OWL is part of the semantic web vision, and recently became a W3 recommendation. It has a large user base, and a lot of tools are being developed for its use[1]. OWL is built upon other W3C recommendations, as is shown in figure 2. The Extensible Markup Language (XML) [4] provides the structure for the documents, but provides no semantics. XML-Schema is a language used to restrict the structure of XML documents and provides data types. Namespaces (NS) provide the means to refer to elements and attributes used in XML documents by associating them with namespaces identified by URIs [3]. The Resource Description Framework (RDF) [2, 12] is a data model inspired by semantic nets. It is possible to formalise this data model in XML, using resources to describe objects and relations between them. RDF provides some simple semantics. RDF Schema [5, 15, 8, 7] extends RDF by introducing constructs to describe classes and properties, and provides the semantics for inheritance hierarchies. The DAML+OIL language was based on RDF Schema. From DAML+OIL the OWL language was derived.

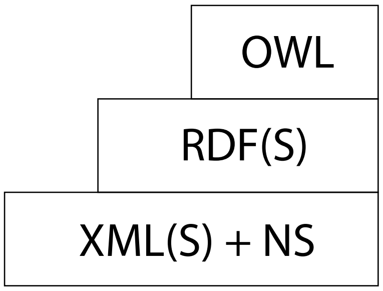

Figure 2: The Layering of W3C Standards.

2.1 OWL File Preamble

As every OWL document is a well-formed XML document, the first line of the OWL file has to be a XML declaration. In the XML declaration the version of XML being used is specified, and optionally in which character encoding the file is saved.

```
<?xml version="1.0"?>
```

After the XML declaration follows a document type declaration (DOCTYPE) in which entities (`&entityname;`) can be defined. These entities can then be used throughout the document as abbreviations of uniform resource identifiers (URI's).

```
<!DOCTYPE rdf:RDF [
    <!ENTITY animal           "http://phyro.mine.nu/animal2.xml#" >
    <!ENTITY colour           "http://phyro.mine.nu/colour.xml#" >
    <!ENTITY owl              "http://www.w3.org/2002/07/owl#" >
    <!ENTITY xsd              "http://www.w3.org/2001/XMLSchema#" >
] >
```

The XML declaration and the document type declaration are the only statements that do not have both an opening tag (`<tagname>`) and a closing tag (`</tagname>`). If a tag does not have any elements nested inside it (`<tagname></tagname>`), the empty element syntax may be used (`<tagname/>`).

[1]For a list of tools, projects and applications see: http://www.w3.org/2004/OWL/#projects

2.2 RDF Opening Tag and Namespaces

The first opening tag in the OWL file is `<rdf:RDF>`, which is closed at the end of the file by `</rdf:RDF>`. The attributes of the opening tag define some XML namespaces. XML namespaces define collections of names, which are identified by an URI. These namespaces are used in XML documents as element types and attribute names, which are defined at the specified URI [3]. Complete URI's in the opening tag can be exchanged for entities defined in the doctype.

```
<rdf:RDF
    xmlns              ="&animal;"
    xmlns:animal       ="&animal;"
    xml:base           ="&animal;"
    xmlns:colour       ="&colour;"
    xmlns:owl          ="http://www.w3.org/2002/07/owl#"
    xmlns:rdf          ="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs         ="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:xsd          ="http://www.w3.org/2001/XMLSchema#">

<!-- The rest of the OWL file -->

</rdf:RDF>
```

The first namespace defines the default namespace, meaning that unprefixed names refer to the current ontology. The second specifies that the current ontology can be referred to using the prefix `animal:`. The third defines the base URL for the ontology. The fourth specifies the namespace of the imported (see below) colour ontology. The fifth through eight namespace declarations state that constructs prefixed with `owl:`, `rdf:`, `rdfs:` and `xsd:` refer to concepts defined in their corresponding namespaces.

2.3 Ontology Meta Data

For purposes of searching for ontologies it is desirable to be able to store meta data about ontologies. This kind of information can be encapsulated inside `owl:Ontology` tags. `owl:label` is used to provide a name for the ontology, but is also (just as `rdfs:comment`) used throughout the ontology to annotate classes and relations. `rdfs:comment` can be used to provide a human readable comment about the ontology. Using the attribute `xml:lang` labels and comments can be provided in multiple languages. `owl:versionInfo` can be used to state the current version of the ontology. `owl:priorVersion` provides an URL to the prior version of the ontology. Finally `owl:imports` is used to import other ontologies. Note that the imported ontology is also mentioned in the `doctype` and `rdf:RDF` tag.

```
<owl:Ontology rdf:about="">
    <rdfs:comment>An example ontology about animals</rdfs:comment>
    <owl:versionInfo>Version 2</owl:versionInfo>
    <owl:priorVersion rdf:resource="http://phyro.mine.nu/animal1.xml"/>
    <rdfs:label xml:lang="en">Animal Ontology</rdfs:label>
    <rdfs:label xml:lang="nl">Dieren Ontologie</rdfs:label>
    <owl:imports rdf:resource="http://phyro.mine.nu/colour.xml"/>
</owl:Ontology>
```

If the capability to store more meta data about the ontology is desired, the Dublin Core Meta Data standard [10] OWL-file can be imported. This meta data ontology defines concepts such as creator, subject, publisher, contributor, type, format, language, etc.

2.4 Classes and Subclasses

The most simple class which can be defined using owl is a class with just a name. To give the class its name `rdf:ID` is used. Implicitly every class is a subclass of `owl:Thing`, which therefore becomes the root node of every class hierarchy. The empty class is defined as `owl:Nothing`. Subclasses are formalised using `rdfs:subClassOf`. To refer to classes or properties named using `rdf:ID`, `rdf:resource` is used.

```
<owl:Class rdf:ID="PhysicalThing"/>

<owl:Class rdf:ID="LivingBeing">
    <rdfs:subClassOf rdf:resource="#PhysicalThing"/>
</owl:Class>

<owl:Class rdf:ID="Animal">
    <rdfs:subClassOf rdf:resource="#LivingBeing"/>
</owl:Class>
```

2.5 Properties and Subproperties

Relations, called properties in OWL, are defined using `owl:ObjectProperty`. In the definition of properties `rdfs:domain` and `rdfs:range` may be specified. The former specifies that the owner of the relation is of the specified class. The latter specifies that the target of the relation is of the specified class. Note that these are not restrictions, but possible inferences. An OWL reasoner may infer the type of the owner and target of the relation if these characteristics are defined. Like classes, properties are ordered in a hierarchy. To state that a property is a subproperty `rdfs:subPropertyOf` is used. `owl:inverseOf` is utilised to specify that a property is the inverse of another property. From property K, its inverse L and the relation K(x,y), it is valid to infer L(y,x). `owl:equivalentProperty` is used to state that two properties have the same values, which does not mean that the properties are the same. Relations between instances of classes and RDF literals or XML Schema data types are specified using `owl:DatatypeProperty`. Commonly used XML Schema data types are `xsd:nonNegativeInteger` and `xsd:string`.

```
<owl:ObjectProperty rdf:ID="consumes">
    <rdfs:domain rdf:resource="#Animal"/>
    <rdfs:range  rdf:resource="#PhysicalThing"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="isConsumedBy">
    <owl:inverseOf rdf:resource="#consumes"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="eats">
    <rdfs:subPropertyOf rdf:resource="#consumes"/>
    <rdfs:domain rdf:resource="#Animal"/>
    <rdfs:range  rdf:resource="#PhysicalThing"/>
</owl:ObjectProperty>

<owl:DatatypeProperty rdf:ID="name">
    <rdfs:domain rdf:resource="#Dog"/>
    <rdfs:range  rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>

<owl:ObjectProperty rdf:ID="hasColor">
    <rdfs:domain rdf:resource="#PhysicalThing"/>
    <rdfs:range  rdf:resource="&colour;Colour"/>
</owl:ObjectProperty>
```

2.6 Characteristics of Properties

OWL provides the possibility to add characteristics to properties which allow additional inferences. A relation can be defined to be transitive by stating that the property is of type `&owl;TransitiveProperty`. Consider a transitive relation R. From the relations R(a,b) and R(b,c) the relation R(a,c) may be inferred. Examples of relations which are transitive are equalities, inequalities and subset relations.

```
<owl:ObjectProperty rdf:ID="isSubsetOf">
    <rdf:type rdf:resource="&owl;TransitiveProperty"/>
    <rdfs:domain rdf:resource="#Set"/>
    <rdfs:range rdf:resource="#Set"/>
</owl:ObjectProperty>
```

Symmetric relations are specified by defining the property as having type `&owl;SymmetricProperty`. Consider a symmetric relation P. From P(x,y) the relation P(y,x) may be inferred. Equality, adjacency, and being married to are symmetric relations.

```
<owl:ObjectProperty rdf:ID="isMarriedTo">
    <rdf:type rdf:resource="&owl;SymmetricProperty"/>
    <rdfs:domain rdf:resource="#Person"/>
    <rdfs:range rdf:resource="#Person"/>
</owl:ObjectProperty>
```

Functional properties specify that there exists only a single individual which is the value of the property. These kind of properties can be defined by specifying that the property is of type `&owl;FunctionalProperty`. From a functional property Q and the relations Q(x,y1) and Q(x,y2), you may infer that y1=y2. Examples of functional properties are: hasFather, hasBirthDate, hasGender.

```
<owl:ObjectProperty rdf:ID="hasFather">
    <rdf:type rdf:resource="&owl;FunctionalProperty"/>
    <rdfs:domain rdf:resource="#Person"/>
    <rdfs:range rdf:resource="#Person"/>
</owl:ObjectProperty>
```

Inverse functional properties specify that the value of a relation is unique for individuals having that property. Stating that the property is of type `&owl;InverseFunctionalProperty` means that the relations is an inverse functional property. From an inverse functional property R and the relations R(x1,y) and R(x2,y), you may infer that x1=x2. Examples of inverse functional relations are: hasSocialSecurityNumber, hasDollarSerialNumber, hasStudentNumber.

```
<owl:ObjectProperty rdf:ID="hasSocialSecurityNumber">
    <rdf:type rdf:resource="&owl;InverseFunctionalProperty"/>
    <rdfs:domain rdf:resource="#Person"/>
    <rdfs:range rdf:resource="#Person"/>
</owl:ObjectProperty>
```

2.7 Classes Defined by Property Restrictions

Restrictions always make use of the `owl:onProperty` construct, specifying the property to which the restriction applies. There should always be a second ingredient used in restrictions, which state the kind of restriction that applies to the property. There are several of these kinds of restrictions: `owl:hasValue` specifies the instance of a class (see section 2.9 for more about individuals) or data type that must be the value of the property. `owl:allValuesFrom` states that the values of the property must be of a certain class, and acts as a universal quantifier. `owl:someValuesFrom` specifies that some of the values of the property must be of a certain class, and acts as a existential quantifier. `owl:cardinality, owl:minCardinality` and `maxCardinality`, specify the exact, minimum and maximum number of values a property must have. All constructs on the same indentation level should be read as having a conjunction in between them. So multiple restrictions have to be written as multiple restriction blocks.

In the next example, the concept polar bear is defined to be a subclass of the concept bear and an anonymous class with certain restrictions. In this case a polar bear must have a white colour. It is perfectly valid to define a class using solely restrictions.

```
<owl:Class rdf:ID="PolarBear">
    <rdfs:subClassOf rdf:resource="#Bear"/>
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasColor"/>
            <owl:hasValue rdf:resource="&colour;White"/>
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
```

The restrictions which belong to a class do not need to be stated directly. It is possible to extend the definition of a class defined in a file somewhere else on the web. To extend a class the keyword `rdf:about` is used to refer to the class.

```
<owl:Class rdf:ID="PolarBear">
    <rdfs:subClassOf rdf:resource="#Bear"/>
</owl:Class>

<owl:Class rdf:about="#PolarBear">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasColor"/>
            <owl:hasValue rdf:resource="&colour;White"/>
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
```

The reason classes can be extended is because a subclass definition in OWL states necessary constraints, but not sufficient. This means that in the previous example you can infer that a polar bear must be white, but not that a bear, which is white is a polar bear. It is possible to state necessary and sufficient restrictions for a class by using `owl:equivalentClass` or `owl:intersectionOf` (section 2.8). Being an instance of a specific class implies that its necessary conditions apply, but an instance fulfilling the necessary and sufficient conditions of a class must be a member of that class.

```
<owl:Class rdf:ID="#PinkThing">
    <owl:equivalentClass>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasColor" />
            <owl:hasValue   rdf:resource="&colour;Pink" />
        </owl:Restriction>
    </owl:equivalentClass>
</owl:Class>
```

2.8 Classes Defined Using Set Theory

Classes can be considered to be sets of individuals. By using set operators on these sets of individuals, other classes can be defined. Like with subclass definitions both classes and anonymous classes can be used.

To define a class as the intersection of other classes and restrictions `owl:intersectionOf` is used. This characteristic requires a `rdf:parseType="Collection"` attribute, which indicates that a list of classes follows. The listing specifies the classes to which the instance must belong and the restrictions it must abide to. Note that `rdf:about` must be used to refer to classes in the intersection, as the definition adds to the definition of the class.

```
<owl:Class rdf:ID="Herbivore">
    <owl:intersectionOf rdf:parseType="Collection">
        <owl:Class rdf:about="#Animal"/>
        <owl:Restriction>
```

```
            <owl:onProperty rdf:resource="#eats"/>
            <owl:allValuesFrom rdf:resource="#Plant"/>
        </owl:Restriction>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#eats"/>
            <owl:minCardinality rdf:datatype="&xsd;nonNegativeInteger">1
            </owl:minCardinality>
        </owl:Restriction>
    </owl:intersectionOf>
</owl:Class>

<owl:Class rdf:ID="Omnivore">
    <owl:intersectionOf rdf:parseType="Collection">
        <owl:Class rdf:about="#Animal"/>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#eats"/>
            <owl:someValuesFrom rdf:resource="#Animal"/>
        </owl:Restriction>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#eats"/>
            <owl:someValuesFrom rdf:resource="#Plant"/>
        </owl:Restriction>
    </owl:intersectionOf>
</owl:Class>
```

A class definition using a union of classes is very similar to the class definition using an intersection. The difference is that `owl:unionOf` is used. To be a member of the defined class, an individual must be a member of one of the classes in the union.

```
<owl:Class rdf:about="#Vertebrate">
    <owl:unionOf rdf:parseType="Collection">
        <owl:Class rdf:about="#Fish"/>
        <owl:Class rdf:about="#Amphibian"/>
        <owl:Class rdf:about="#Reptile"/>
        <owl:Class rdf:about="#Bird"/>
        <owl:Class rdf:about="#Mammal"/>
    </owl:unionOf>
</owl:Class>

<owl:Class rdf:about="#PhysicalThing">
    <owl:unionOf rdf:parseType="Collection">
        <owl:Class rdf:about="#LivingBeing"/>
        <owl:Class rdf:about="#NonLivingThing"/>
    </owl:unionOf>
</owl:Class>
```

To indicate that the individuals of a class are exactly all the members which do not belong to another class, the `owl:complementOf` construct is used. This construct can also be used in combination with restrictions, to specify all the individuals which do not fulfil the restriction.

```
<owl:Class rdf:ID="NonLivingThing">
    <rdfs:subClassOf rdf:resource="#PhysicalThing"/>
    <owl:complementOf rdf:resource="#LivingBeing"/>
</owl:Class>

<owl:Class rdf:ID="Vegetarian">
    <owl:intersectionOf rdf:parseType="Collection">
        <owl:Class rdf:about="#Animal"/>
        <owl:complementOf>
            <owl:Restriction>
                <owl:onProperty rdf:resource="#eats"/>
                <owl:hasValue rdf:resource="#Animal"/>
            </owl:Restriction>
        </owl:complementOf>
    </owl:intersectionOf>
</owl:Class>
```

To specify that members of a class cannot be simultaneously members of another class the `owl:disjointWith` construct is used.

```
<owl:Class rdf:about="#Animal">
    <owl:disjointWith rdf:resource="#Plant" />
    <owl:disjointWith rdf:resource="#Bacteria" />
    <owl:disjointWith rdf:resource="#Fungus" />
    <owl:disjointWith rdf:resource="#Virus" />
</owl:Class>
```

As mentioned in section 2.7 using boolean combinations (and `owl:equivalentClass`) it is possible to state necessary and sufficient conditions. In the next example a polar bear is defined as the intersection of the class bear and things which are white. In contrast to the polar bear definition in 2.7 it is now valid to infer from an individual which is a bear and is white that it is a polar bear.

```
<owl:Class rdf:ID="PolarBear">
    <owl:intersectionOf rdf:parseType="Collection">
        <owl:Class rdf:resource="#Bear"/>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasColor"/>
            <owl:owl:hasValue rdf:resource="&colour;White/>
        </owl:Restriction>
    </owl:intersection>
</owl:Class>
```

2.9 Individuals

The definition of classes sometimes depends the existence of individuals. For example classes can be defined as things with properties having specific values, or as the enumeration of the individuals which belong to the class. Therefore, *"in OWL the term ontology has been broadened to include instance data (...)"* [19].

Instances of classes and instances of properties are introduced using the names of those classes and properties. Another way is to introduce an `owl:Thing` and specify its type using `rdf:type`.

```
<Bear rdf:ID="JohnnyBear">
    <eats rdf:resource="#MikeGoldFish"/>
    <hasColor rdf:resource="&colour;White"/>
</Bear>

<owl:Thing rdf:ID="WilliamBear">
    <rdf:type rdf:resource="#Bear"/>
    <eats rdf:resource="#MikeGoldFish"/>
    <hasColor rdf:resource="&colour;White"/>
</owl:Thing>
```

As OWL does not have a unique name assumption, the two bears in the previous example are not necessarily different. It is possible to say that the two are the same using `owl:sameAs`.

```
<Bear rdf:about="#WilliamBear">
    <owl:sameAs rdf:resource="#JohnnyBear" />
</Bear>
```

It is also possible to say that the two bears are different using `owl:differentFrom`. It would be rather unwieldy to specify for each individual that it is different from the others. Therefore, it is possible to state that all individuals in a set are distinct using `owl:allDifferent` in combination with `owl:distinctMembers`.

```
<Bear rdf:about="#WilliamBear">
    <owl:differentFrom rdf:resource="#JohnnyBear" />
</Bear>

<owl:allDifferent>
    <owl:distinctMembers rdf:parseType="Collection">
        <Bear rdf:about="WilliamBear" />
        <Bear rdf:about="JohnnyBear" />
    </owl:distinctMembers>
</owl:allDifferent>
```

2.10 Classes Defined by Enumerations of Individuals

OWL provides the possibility to define a class by specifying its members. Such an enumeration is done using the `owl:oneOf` construct.

```
<owl:Class ID="Gender">
    <owl:oneOf rdf:parseType="Collection">
        <owl:Thing rdf:about="#Male"/>
        <owl:Thing rdf:about="#Female"/>
    </owl:oneOf>
</owl:Class>
```

2.11 The Three Species of OWL

OWL consists of three increasingly expressive sublanguages. Every ontology written in a less expressive sublanguage is a legal ontology in the more expressive sublanguages.

OWL Lite supports only the values 0 and 1 for the cardinality constraints `owl:minCardinality` and `owl:maxCardinality`.

OWL DL is a description logic, as OWL DL can be mapped to the $\mathcal{SHIQ}$ logic [11]. The idea is that OWL DL maintains the maximum expressiveness while maintaining computational completeness (all inferences are guaranteed to be computed), and decidability (the inferences will be computed in finite time). This is an excellent example of the tradeoff between expressiveness and decidability [13]. The distinctive restriction in OWL DL is type separation. It is forbidden to treat a a class as an instance, and a property as an instance of a class. A disadvantage of OWL Lite and OWL DL is that backwards compatibility with RDF is lost. This means that not every valid RDF file is a valid OWL Lite or DL file.

OWL Full has maximum expressiveness and is backwards compatible with RDF, but there are no computational guarantees. There is no type separation restriction, so classes can be treated as individuals, and properties as instances of properties. Data type properties can be given the property characteristic `owl:InverseFunctionalProperty`.

2.12 Conclusions

This section has summarised the Web Ontology Language and answered the question: *"What constructs are available in the Web Ontology Language?"* Constructs have been presented to define class and property hierarchies, add characteristics allowing inferences to properties, add restrictions to classes, define classes using set theory, and stating necessary and sufficient conditions.

3 A Qualitative Reasoning and Modelling Ontology

The formalisation of the QR domain starts with the ordering of the vocabulary in a class hierarchy. Figure 3 shows the taxonomy of the QR model ingredients. The taxonomy shows both the QR concepts and relations, as the latter are reified (treated as classes). The top node of is called QualitativeModelIngredient, as every class is a possible ingredient of a qualitative model. The model ingredients are divided into the sets BuildingBlock and Aggregate. The former describes separate model ingredients, while the latter describes collections of related model ingredients. The aggregate concepts ModelFragment and Scenario are described in section 7. Qualitative models describe both the structural and the behavioural aspects of systems. Therefore, the building blocks are subdivided into the sets Structural Building Blocks (section 4), Behavioural Building Blocks (section 5), and AssumptionType (section 8.1). Assumptions are considered to be separate, as they do not describe inherent aspects of the system.

One of the most difficult problems when developing an ontology is finding proper names for the defined concepts. Whenever a vocabulary concept is described in this document, its position in the hierarchy will be discussed along with the naming (if it differs from the standard QR vocabulary). The formalisation of the hierarchy in OWL, however, is straighforward. The classes are defined by giving them an id, label, comment, and specifying their superclasses. As OWL assumes that the sets which the classes model overlap, the siblings on each level have been specified as being disjoint. In the rest of this chapter, the definitions of the concepts are extended (see section 2.7) by adding restrictions. The OWL code for the qualitative vocabulary can be found in appendix B.1.

3.1 Ontology Validation

In order to check if the ontologies defined in this chapter are correct three things were done. Firstly, and most importantly, the definitions of the QR concepts were discussed with QR researchers. This was done to make sure the ontology developer perspective is the same as the users of the ontology. Secondly, the consistency of the generic ontology was ensured by repeatingly testing it using an OWL reasoner[2]. Finally, a parser was developed which converts the knowledge representation of Garp3 to a knowledge representation in OWL and back. The domain ontologies which this parser creates refer to the vocabulary defined in the generic ontology. These models are used to check if the defined vocabulary is consistent. Using graphical OWL editors[3][4], an OWL reasoner, and some validators[5][6] the domain models (which include the generic ontology via the web) are analysed for their well-formedness and consistency.

3.2 Notation

The notation used in the coming chapters is originates from the restriction editor in Protege OWL, and is shown in Table 1.

3.3 Model Ingredient

Every model ingredient in the qualitative reasoning vocabulary is a subclass of the QualitativeModelIngredient class. Almost all instances of model ingredients used in model fragments and scenarios have to be visualised in Garp3. Furthermore, they can also be hidden. This is modelled using three datatype properties. The instances of the model ingredient class can only have one x and one y position, and the ingredient is either hidden or visible. As not every model ingredient has a specific position which needs to be stored (because the position is indicated by other ingredients), filling

[2]Racer: http://www.racer-systems.com/
[3]Protege: http://protege.stanford.edu/
[4]Triple 20: http://www.swi-prolog.org/packages/Triple20/
[5]Wonderweb: http://phoebus.cs.man.ac.uk:9999/OWL/Validator
[6]BBN: http://owl.bbn.com/vowlidator/

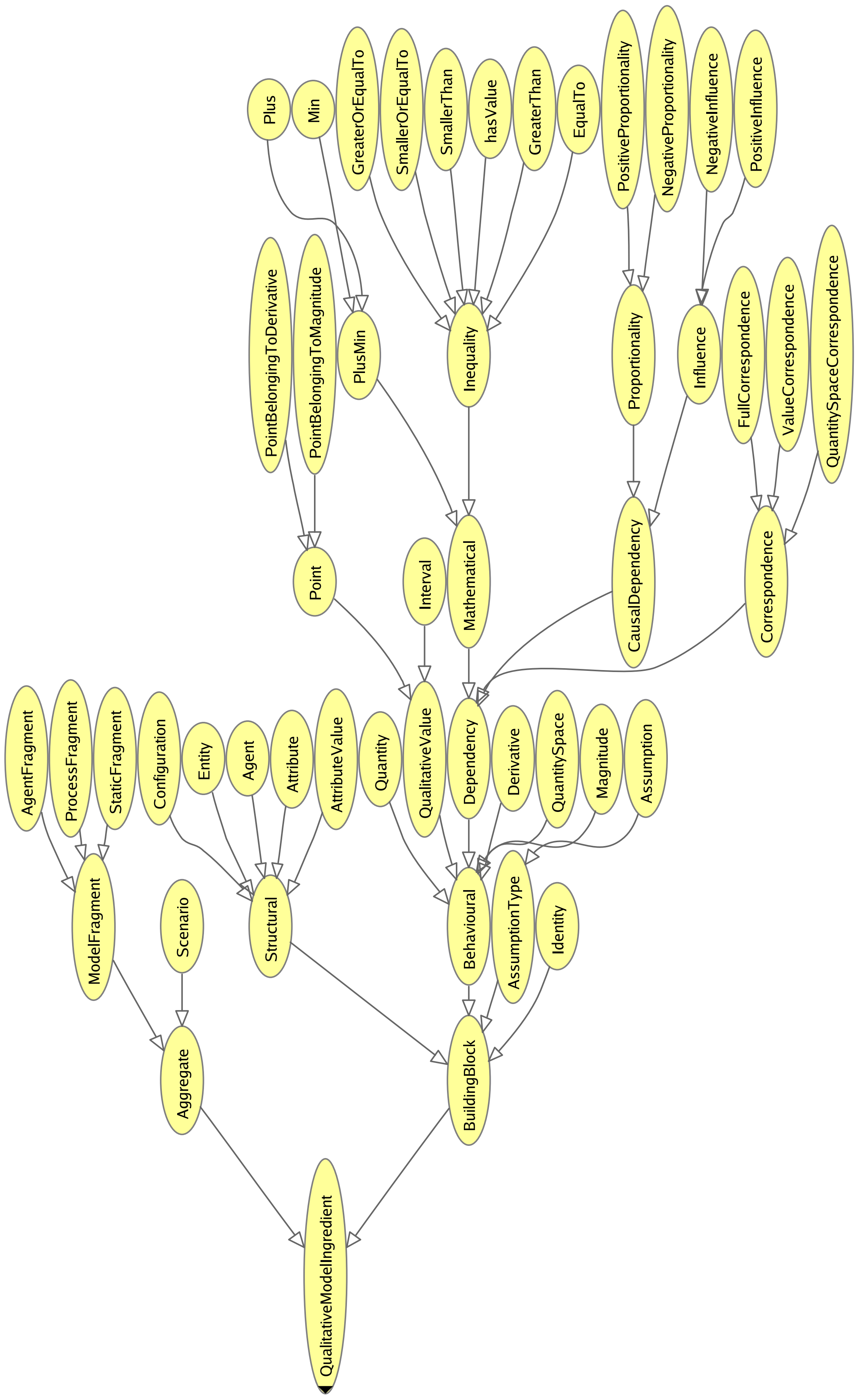

Figure 3: The QR ingredient taxonomy defining the QR vocabulary.

OWL Element	Symbol	Example	Meaning
allValuesFrom	$\forall$	$\forall\ Person\ Human$	Every person is human
someValuesFrom	$\exists$	$\exists\ Person\ Male$	At least one person is male
hasValue	$\ni$	$isConsistent \ni true$	The consistent property must be true
cardinality	$=$	$hasWings = 2$	There must be exactly 2 wings
minCardinality	$\geq$	$hasEngines \geq 2$	There must be at least 2 engines
maxCardinality	$\leq$	$hasPassengers \leq 300$	There may be at most 300 passengers
complementOf	$\neg$	$\neg\ Male$	Everything not Male
intersectionOf	$\sqcap$	$Human \sqcap Male$	Every Human which is Male
unionOf	$\sqcup$	$Male \sqcup Female$	Anything that is either Male or Female
enumeration	$\{\ldots\}$	$\{John\ Mike\}$	The individuals John or Mike

Table 1: The equivalent OWL notation used in this section.

these values is not always necessary. Therefore, instead of cardinality restrictions, maxcardinality restrictions are used. The OWL code for the QualitativeModelIngredient class can be found in appendix B.2.

Necessary QualitativeModelIngredient restrictions:

$has\_xposition\_on\_screen \leq 1$

$has\_yposition\_on\_screen \leq 1$

$is\_hidden\_on\_screen \leq 1$

4 Structural Building Blocks

This section describes structural ingredients set of the qualitative model ingredients, which are used to describe the structure of a system. This class encompasses the concepts (1) *entities*, (2) *agents*, (3) *configurations*, (4) *attributes* and (5) *attribute values*.

4.1 Entities and Agents

Entities describe the objects which exist in a system. Agents are very similar in the kind of relations they can take part in, but are used to model 'outside forces' on the system. They are both structural ingredients, and are alike in the kinds of relations they can take part in. It is possible for entities and agents to have attributes (section 4.3) and quantities (section 5.1) as is shown in figure 4. These necessary restrictions are shown below. Entities and agents can have configuration relations with other entities and agents, which are described in section 4.2. The specific way in which the configurations are formalised has to do with the reusability of reificated relations [14]. The OWL formalisation of the configuration restrictions can be found in appendix B.4.1.

Necessary Entity and Agent restrictions:

$\forall\ hasAttribute\ Attribute$

$\forall\ hasConfiguration\ (Configuration\ \sqcap$

$\quad (\exists\ hasConfigurationTarget\ (Entity\ \sqcup\ Agent)))$

$\forall\ hasQuantity\ Quantity$

Developers of qualitative models can define their own entities and agents in a subtype hierarchy. These entities and agent definitions are stored in the *domain ontology*. The formalisation of these hierarchies in OWL is similar to the formalisation of the QR vocabulary hierarchy (section 3). Again, the classes are defined with their respective superclasses, and disjointness axioms have to be added to indicate that individuals of classes cannot be members of the sibling classes. The top nodes of these hierarchies are the general *entity* and *agent* concepts defined in the *generic ontology*. Therefore, every model imports the generic ontology, and refers to these concepts though the correct namespace. An example of the formalisation of the entity and agent hierarchies, generated by the model parser, can be found in appendix C.1.

4.2 Configurations

Configurations have exactly one owner and one target, therefore the hasConfigurationTarget and hasConfigurationOwner relations are defined as having cardinality 1. By defining inverse relations the owner belonging to the configuration, and the configuration which targets an entity or agent can be derived. The OWL definitions can found in appendix B.4.

Necessary Configuration Restrictions

$hasConfigurationTarget\ =\ 1$

Developers of qualitative models can define their own configurations. In contrast with entities and agent, configurations are not arranged in a taxonomy. As a result every configuration is a subclass of the concept *configuration* defined in the generic ontology. An example of some configuration definitions in OWL can be found in appendix C.2

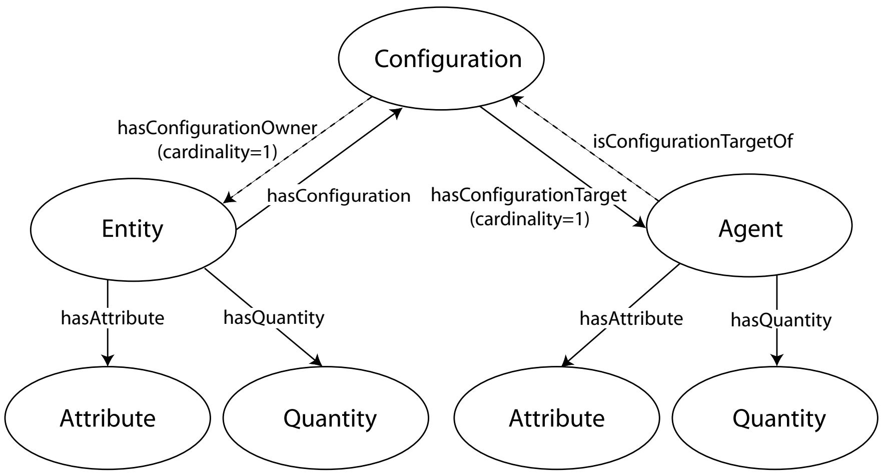

Figure 4: The possible relations of *entities* and *agents*. Note that both the entity may be changed to agent, and the agent into entity.

4.3 Attributes and Attributes Values

Attributes describe the features of entities and agents that do not change gradually. Models can define their own attributes, which consist of an attribute name, and its possible values. Attributes are considered to be properties of entities and agents, and should have exactly one attribute value. As OWL does not make a distinction between relations between objects and properties of objects[7], a pattern has to be used to formalise properties in OWL. An existing pattern to model property values uses an enumeration of individuals [18].

Property Values as an enumeration of individuals In the enumeration pattern the values of a property are considered to be a set of individuals, as shown in figure 5. As a result the values are unique in the ontology, and objects with the property all refer to a value from the same set of individuals. It is necessary to explicitly state that the values are different using the owl:differentFrom statement, as OWL does not make the Unique Name Assumption (two individuals with different names are not necessarily different objects). The formalisation of this pattern in OWL can be found in appendix A.1.1.

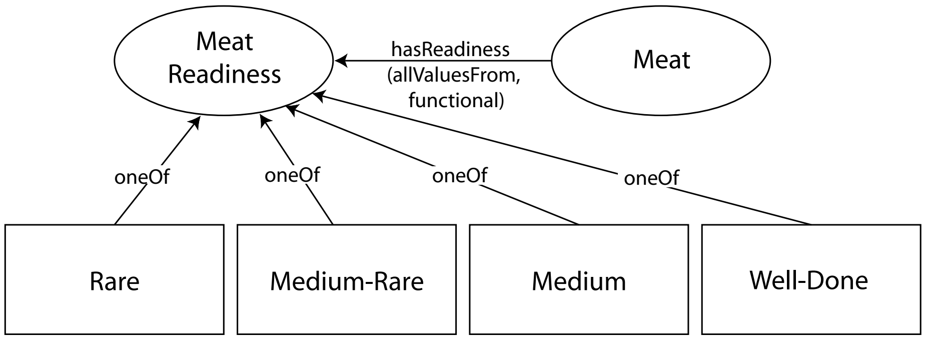

Figure 5: Values as an enumeration of individuals.

Back to the formalisation of attributes. The attribute value is defined to be an enumeration

[7]Not to be confused with ObjectProperties in OWL

of individuals (see figure 6). This is possible as attribute values do not have a position on the screen which has to be stored. This allows the same value set to be used for each instance of an attribute. Attribute relations are a typical example of reification pattern 1. The attribute itself is an independent object, while the attribute values are dependent on the existence of the attribute. Since an attribute always has a relation with an attribute value, the semantics are stored in the attribute class instead of in the classes having attributes. The formalisation of the attribute and attribute value concepts in OWL is shown in appendix B.5.

Attribute Restrictions
$\forall\ hasAttributeValue\ AttributeValue$
$\forall\ hasAttributeValue\ =\ 1$

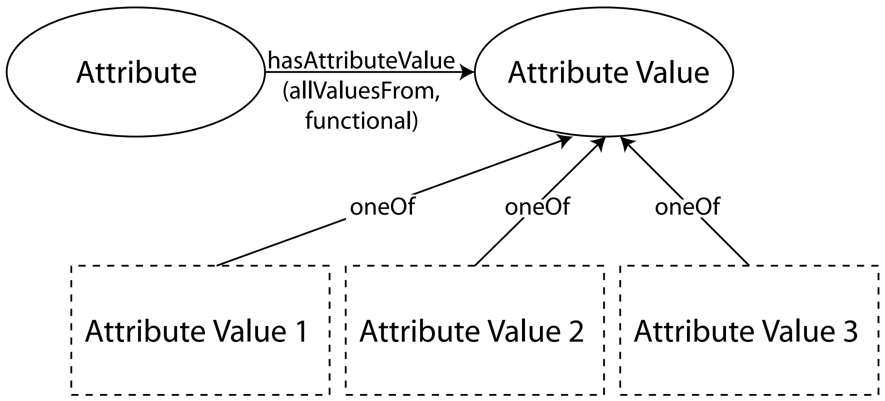

Figure 6: The attribute components and its relations.

The attributes defined in the domain ontology have some further restrictions. The attribute is prohibited to have any other attribute value, other than an instance of the associated attribute value class. The attribute value class is defined as the enumeration of the possible values. Examples of attribute and attribute value definitions in a model context can be found in appendix C.4.

OpenOrClosed Restrictions
$\forall\ hasAttributeValue\ OpenOrClosedValue$
OpenOrClosedValue Restrictions
$\{Open, Closed\}$

5 Behavioural Building Blocks

Behavioural ingredients describe the model ingredients used to describe the behaviour of a system. They include: (1) *quantities*, (2) *Magnitudes*, (3) *derivatives*, (4) *quantity spaces*, (5) *qualitative values*, and (6) *dependencies*. Qualitative values can be further divided into *points* and *intervals*.

5.1 Quantities, Magnitudes and Derivatives

Quantities describe the changeable features of entities and agents. These behavioural ingredients consist of exactly one magnitude and exactly one derivative (see figure 7). Both the magnitude and the derivative have exactly one quantity space. The formalisation of the quantity, magnitude and derivative restrictions in OWL is shown in appendix B.6.

The magnitude is actually the "value" of the quantity, while the derivative describes its trend. Forbus uses the term *amount* to refer to the "value" (magnitude), and magnitude to refer to the value of the magnitude or derivative [6]. The term amount is not used in our approach, because it is ambiguous. Next to Forbus' use, it could also be used to refer to the quantity "amount". Another alternative is the term value, but that word has the problem that the qualitative values within a quantity space are also referred to as values. Magnitude does not have these problems, and is an appropriate name in this context. Note that Garp3 does not distinguish a separate magnitude concept from a quantity. In that context, the relations meant for the magnitude are established on the quantity. Quantities can have proportionality and influence relations (section 6.3.1). Magnitudes and Derivatives can participate in (in)equality relations (section 6.1.2) and Calculi relations (section 6.1.1).

Necessary Quantity Restrictions:

$\forall\ hasMagnitude\ Magnitude$

$hasMagnitude\ =\ 1$

$\forall\ hasDerivative\ Derivative$

$hasDerivate\ =\ 1$

Necessary Magnitude Restrictions:

$\forall\ hasQuantitySpace\ QuantitySpace$

$hasQuantitySpace\ =\ 1$

Necessary Derivative Restrictions:

$\forall\ hasQuantitySpace\ QuantitySpace$

$hasQuantitySpace\ =\ 1$

Qualitative model builders can define their own quantities. A quantity can have a number of allowed quantity spaces (section 5.2). This information has to be stored in the specific quantity concept in the domain ontology. The formalisation in OWL is done using a restriction. The quantity has a magnitude, which has a quantity space relation with one of the quantity spaces in a union. The formalisation of a quantity space in OWL is shown in appendix B.6.

Necessary Flow Restrictions:

$\exists\ hasMagnitude(Magnitude\ \sqcap$

$(\forall\ hasQuantitySpace(Minimumnegativezeropositivemaximum\ \sqcup$

$Negativezeropositive\ \sqcup\ Zeropositivemaximum)))$

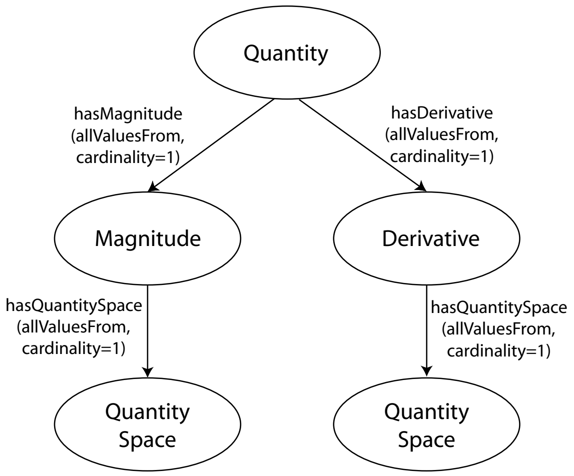
Quantity
hasMagnitude (allValuesFrom, cardinality=1)
hasDerivative (allValuesFrom, cardinality=1)
Magnitude
Derivative
hasQuantitySpace (allValuesFrom, cardinality=1)
hasQuantitySpace (allValuesFrom, cardinality=1)
Quantity Space
Quantity Space

Figure 7: The quantity has one magnitude and one derivative, which both have quantity spaces.

5.2 Quantity Spaces and Qualitative Values (point/interval)

The quantity space is a behavioural ingredient which defines the possible qualitative values a quantity can have. A quantity space consists of at least one qualitative value. These values are also behavioural ingredients, and can be either a point or an interval. The quantity space describes a total order, meaning that a magnitude or derivative can only change to a value directly above or below its current value. Quantity spaces can participate in a correspondence relations (section 6.2).

Necessary QuantitySpace Restrictions
$\forall\ hasQualitativeValue\ QualitativeValue$
$hasQualitativeValue\ \geq\ 1$

As each qualitative value in a model fragment can participate in (in)equality relations, it is impossible to formalise them as an enumeration of individuals. It is more graceful to model each value which can participate in a relation as a separate individual. Therefore, "the property values as a set of individuals" pattern cannot be used. Another existing pattern which is more appropriate formalises each value as a class [18].

Property Values as classes Values of properties can be thought of as a set of subclasses forming a parent class (see figure 8). This class binds all the possible value classes of the property using the owl:unionOf construct. It is necessary to explicitly state that the subclasses are disjoint, as otherwise a property could have two values at the same time (because they are the same, i.e. the value is an instance of both subclasses). In contrast with the 'property values as individuals' pattern, values are not unique, but a new instance of the value is created for each property instance. The OWL code corresponding to this pattern can be found in appendix A.1.2. A problem with this pattern is that the property values do not have an explicit order, which is needed for quantity spaces. An existing pattern which might solve this problem is to model the values as a sequence in a list.

Property Values as a Sequence in a List This pattern is described in the n-ary relations document of the Semantic Web Working Group [16]. If the possible values of a property have a strict sequence, the previous patterns are not expressive enough, as they do not enforce an ordering. A possible solution to this problem is to model the values as a list. Unfortunately, using rdf:List in an OWL ontology causes it to become OWL Full. A solution is to model a list in OWL (see figure 9), which is then used to store values.

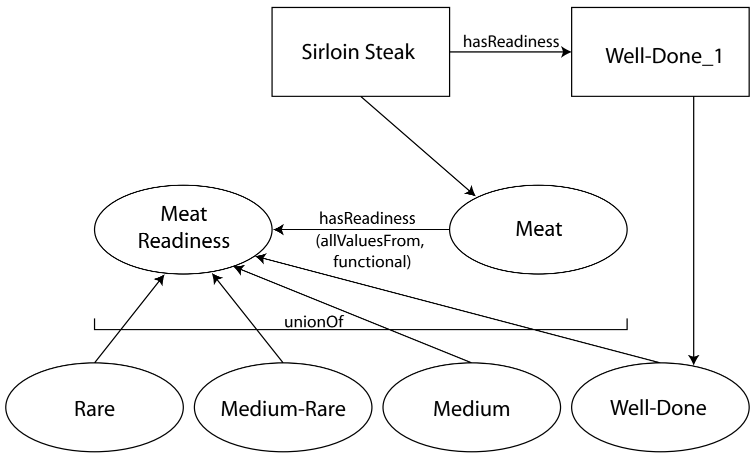

Figure 8: Property values as the instances of classes which form a union.

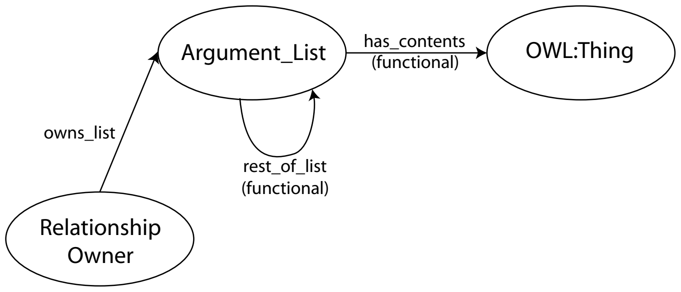

Figure 9: The definition of a list in OWL.

A list consists of a number of arguments, each pointing to the next item in the list. Each argument item has as has_content relation with the an object. The list pattern can be used in combination with both the "property values as an enumeration of individuals", and the "property values as a set of classes" patterns. To formalise a quantity space the latter pattern must be used. The values are ordered in a list to model their sequence, as is shown in figure 10. The OWL formalisation of this pattern can be found in appendix A.1.3.

However, there are two problems with the list pattern. Firstly, a list has no ontological meaning, as it is a data structure. A list with the cities Amsterdam, Brussel, and Paris has little meaning. They could indicate a travel route, cities with the around the same amount of inhabitants, or something else entirely. Secondly, it is awkward to determine the owner of a list from one of the possible values. One has to "reason through" all the values. In order to solve these problems a new pattern had to be developed; the ontological sequence.

Ontological Sequence In order to formalise a quantity space the ordering of the values has to be made explicit. The quantity space is connected with its points and intervals using containsQualitativeValue relations, as is shown in figure 11. The ordering is established by using inequalities (section 6.1.2). Each consecutive value in the order must have another type than the previous

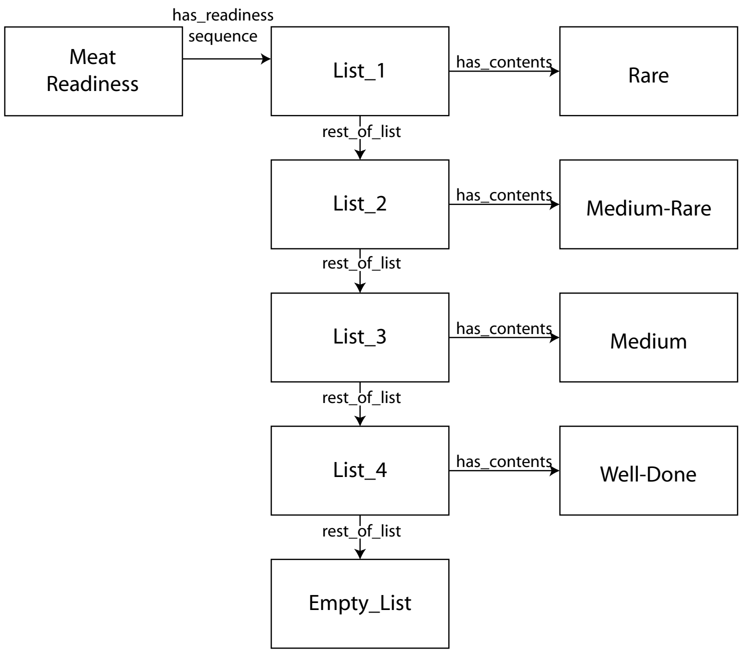

Figure 10: The application of a list in OWL.

one. For that reason, the (in)equality restrictions are formalised in the intervals. This still allows connecting two points, but that is necessary for the other (in)equality relations (section 6.1.2). The point Zero is universal among quantity spaces, and is therefore defined in the generic ontology. The formalisation of the quantity restrictions can be found in appendix B.7. Qualitative model builders can specify their own quantity spaces. These have the form shown in figure 11. An OWL formalisation of an example quantity space can be found in appendix C.5.

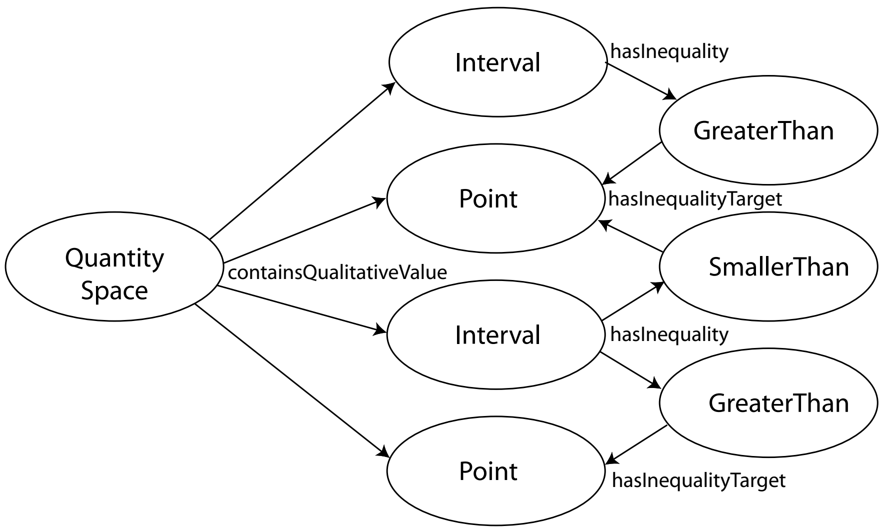

Figure 11: The formalisation of a quantity space and its values using inequalities.

Necessary Quantity Restrictions

$\forall\ containsQualitativeValue\ QualitativeValue$

$containsQualitativeValue\ \geq\ 1$

Necessary NegativeZeroPositive Restrictions

$\exists\ containsQualitativeValue(Positive \sqcap$

$\quad (\exists\ hasInequality(GreaterThan\ \sqcap\ (hasInequalityTarget\ \ni\ Zero))))$

$\exists\ containsQualitativeValue(Negative\ \sqcap$

$\quad (\exists\ hasInequality(SmallerThan\ \sqcap\ (hasInequalityTarget\ \ni\ Zero))))$

$containsQualitativeValue\ \ni\ Zero$

6 Behavioural Dependencies

Dependencies are the possible relations between the behavioural ingredients of a system. They are used to model the processes causing change, constrain what changes can occur, and how these changes occur. The dependency ingredients include: (1) the *causal dependencies: proportionality* and *influence*, (2) *correspondences*, and (3) the *mathematical dependencies: calculi* and all *(in)equalities*.

6.1 Mathematical Dependencies

6.1.1 Calculi

Plus and min relations are tertiary relations which add or substract two values, and indicate that a magnitude, derivative or point is equal to the result. Two different calculi relations may be distinghuised. In the first, only magnitudes and point belonging to magnitudes may participate in the relation (see figure 12). In the second only derivatives may participate in the relation (see figure 13). Both calculi relations should have exactly one left-handside and one right-handside. Any number of inequalities may be placed from the calculi relations to valid targets. The restrictions for the first calculus relation are formalised in both the PointBelongingToMagnitude class (see section 6.1.2) and the Magnitude class. The restrictions for the second is formalised in the derivative class. The OWL formalisation of calculi relations can be found in appendix B.11.

PointBelongingToMagnitude Restrictions:

$$\begin{aligned}&\forall\ isLefthandSideOf(PlusMin \sqcap (hasLefthandSide = 1) \sqcap\\ &\quad (\forall\ hasRighthandSide\ (Magnitude \sqcup PointBelongingToMagnitude)) \sqcap\\ &\quad (hasRighthandSide = 1) \sqcap\\ &\quad (\forall\ hasInequality(Inequality \sqcap\\ &\quad (\forall\ hasInequalityTarget(Magnitude \sqcup PointBelongingToMagnitude)))))\end{aligned}$$

Magnitude Restrictions:

$$\begin{aligned}&\forall\ isLefthandSideOf(PlusMin \sqcap (hasLefthandSide = 1) \sqcap\\ &\quad (\forall\ hasRighthandSide(Magnitude \sqcup PointBelongingToMagnitude)) \sqcap\\ &\quad (hasRighthandSide = 1) \sqcap\\ &\quad (\forall\ hasInequality(Inequality \sqcap\\ &\quad (\forall\ hasInequalityTarget(Magnitude \sqcup PointBelongingToMagnitude)))))\end{aligned}$$

Derivative Restrictions:

$$\begin{aligned}&\forall\ isLefthandSideOf(PlusMin \sqcap (hasLefthandSide = 1) \sqcap\\ &\quad (\forall\ hasRighthandSide\ Derivative) \sqcap\\ &\quad (hasRighthandSide = 1) \sqcap\\ &\quad (\forall\ hasInequality(Inequality \sqcap\\ &\quad (\forall\ hasInequalityTarget\ Derivative))))\end{aligned}$$

6.1.2 Inequalities

Inequalities are dependencies which can be used by a lot of ingredients. They indicate the difference between, or equality of values. The same reificated (in)equality relations SmallerThan, SmallerOrE-

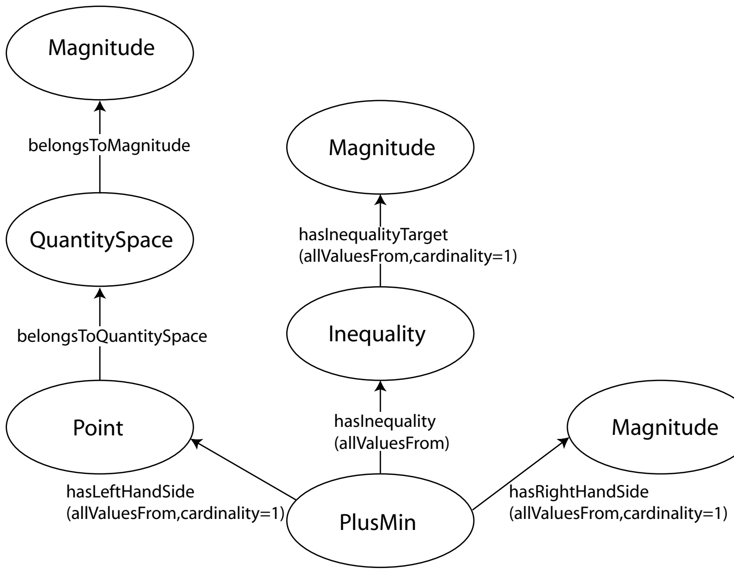

Figure 12: The possible Plus and Min relations between magnitudes and points belonging to magnitudes. Note that every magnitude and the point belonging to the magnitude could be interchanged.

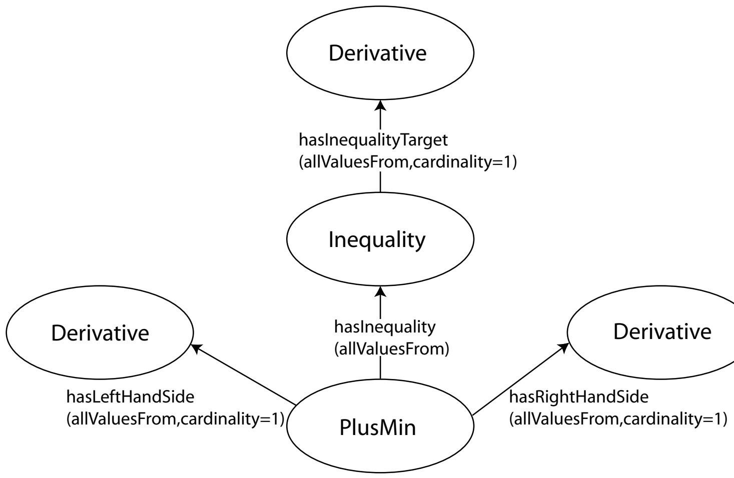

Figure 13: The possible Plus and Min relations between derivatives.

qualTo,EqualTo,GreaterOrEqualTo and GreaterThan are reused for each possible domain/range combination. This reuse is possible as the restrictions are formalised in the classes owning the relations (as mentioned before check [14] for the reasons).

The first (in)equality relation which is described are the inequalities between points. These relations are only valid if both points belong to either a magnitude or a derivative. As it is undesirable to have to distinguish between these type of points in a qualitative model, a pattern is developed to formalise these specific restrictions.

Restrictions for Classes Meeting Specific Conditions There are cases when the relations of an individual have to be restricted depending on the relations that individual has. For example, a workspace meant for a desktop computer should also contain a monitor, but a workspace meant for a laptop does not. Because of the option between a laptop or computer necessity of having a monitor cannot be modelled as a condition in the workspace class.

This problem can be solved by defining a new class workspaceWithDesktop in which necessary and sufficient, and necessary conditions are combined. Recall that necessary conditions are restrictions a consistent individual has to adhere to. Being an individual (or subclass) of such a class implies that its conditions apply ($class \implies conditions$). Necessary and sufficient conditions should be read as an equivalence relation. Fulfilling the conditions means the individual is an instance of the class, and being an individual of the class means the conditions apply ($class \iff conditions$).

The class WorkspaceWithDesktop (see figure 14) has "being of the class Workspace" and "having a desktop as a computer" as necessary and sufficient conditions, meaning that individuals fulfilling these conditions are classified as being a WorkspaceWithDesktop. Individuals of this class have to fulfil the necessary conditions, in this case containing a monitor. As shown combining necessary and sufficient, and necessary conditions allows modelling additional restrictions to individuals adhering to certain conditional relations.

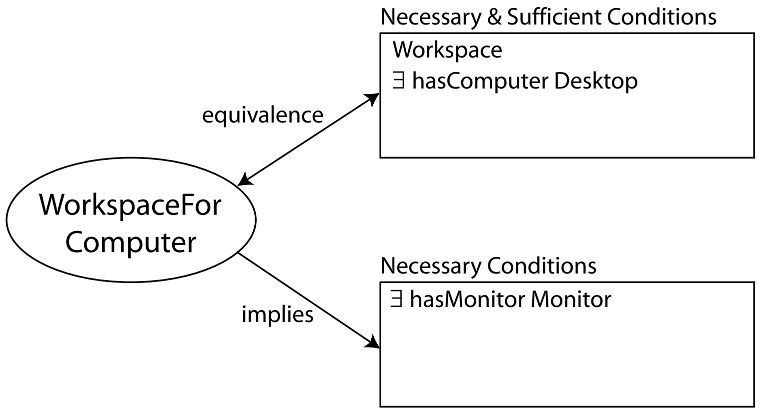

Figure 14: Combining necessary and sufficient, and necessary conditions.

Inequalities from Points Belonging to Magnitudes or Derivatives Points belonging to magnitudes (derivatives) can be classified as such by combining necessary, and necessary and sufficient conditions. Points belonging to magnitudes (derivatives) can have (in)equality relations with other points belonging to magnitudes (derivatives). Using the presented pattern necessary and sufficient conditions are stated to classify points, after which the necessary restrictions on the (in)equality apply. The range of the (in)equality should not be a value in the same quantity space, but this is not expressible in OWL. The restrictions can be seen in figures 15 and 16. The formalisation in OWL is shown in appendix B.10.1.

PointBelongingToMagnitude Necessary and Sufficient Conditions:
$\exists\ belongsToQuantitySpace\ (QuantitySpace\ \sqcap\ (\exists\ isQuantitySpaceOf\ Magnitude))$
PointBeloningToMagnitude Necessary Conditions:
$\forall\ hasInequality\ PointBelongingToMagnitude$

PointBelongingToDerivative Necessary and Sufficient Conditions:
$\exists\ belongsToQuantitySpace\ (QuantitySpace\ \sqcap\ (\exists\ isQuantitySpaceOf\ Derivative))$
PointBeloningToDerivative Necessary Conditions:
$\forall\ hasInequality\ PointBelongingToDerivative$

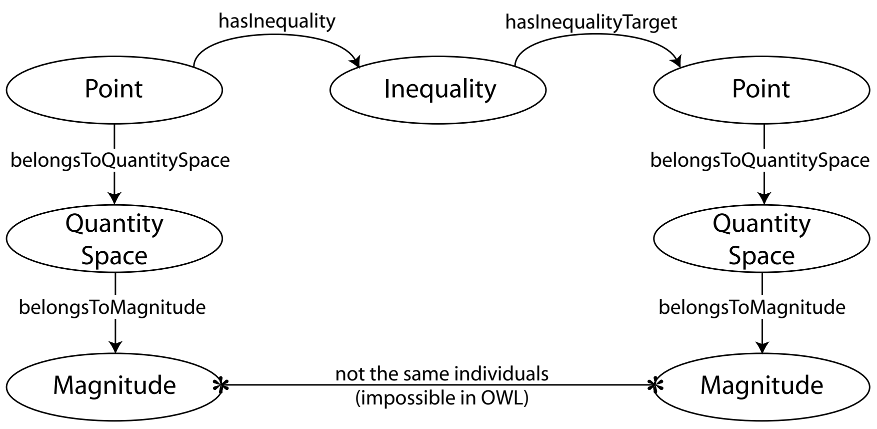

Figure 15: Points belonging to magnitudes can only have (in)equality relations with points belonging to magnitudes.

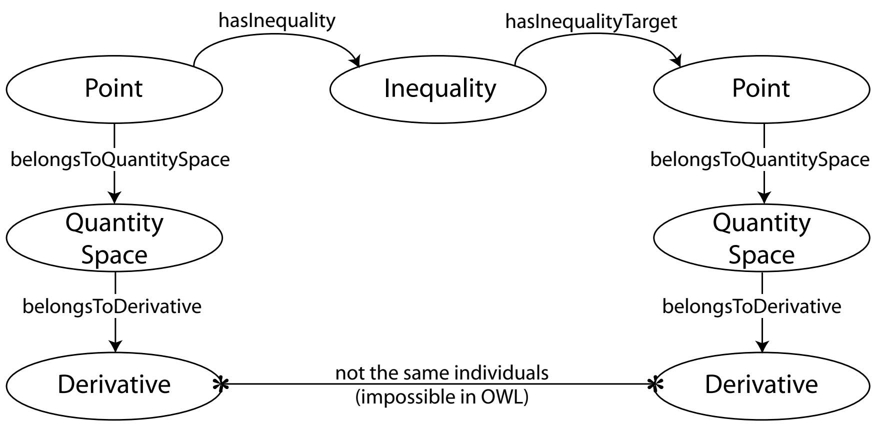

Figure 16: Points belonging to derivatives can only have (in)equality relations with points belonging to derivatives.

Inequalities originating from Magnitudes and Derivatives Magnitudes (derivatives) can have (in)equality relations with other magnitudes (derivatives) (figure 17 and 18). Again non-reflexivity is impossible to formalise. Magnitudes (derivatives) can also have (in)equality relations with qualitative values from their own quantity space (figure 19 and 20). As it is not possible to refer to the individual to which the restriction applies, this restriction is also not formalised. As a result, undesirable (in)equality relations can be formalised, while the ontology remains consistent. Firstly, reflexive inequalities can be described, which are not valid in Garp3. Secondly, inequalities from magnitudes (derivatives) to qualitative values of other magnitudes (derivatives) can be described, which are also not allowed in Garp3. The formalisation in OWL is shown in appendix B.10.2.

Magnitude Necessary Conditions:
$\forall\ hasInequality(Inequality\ \sqcap\ (\forall\ hasInequalityTarget\ (Magnitude\ \sqcup\ Point)))$
Derivative Necessary Conditions:
$\forall\ hasInequality(Inequality\ \sqcap\ (\forall\ hasInequalityTarget\ (Derivative\ \sqcup\ Point)))$

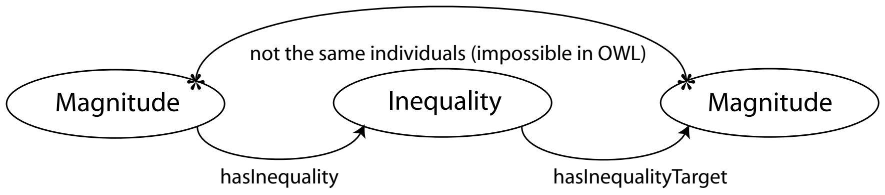

Figure 17: A valid (in)equality relations between two magnitudes.

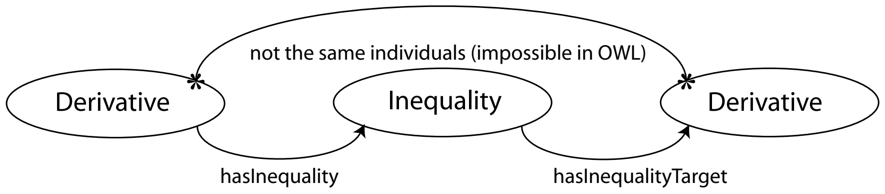

Figure 18: A valid (in)equality between two derivatives.

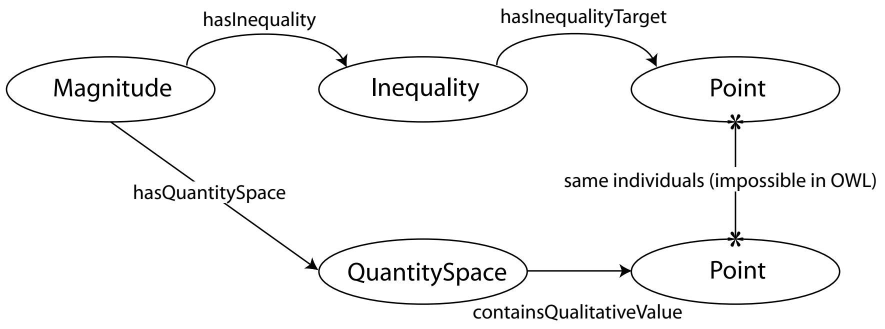

Figure 19: An (in)equality relations between a magnitude and a point.

Inequalities Used to Model the Total Order in Quantity Spaces The inequalities used to model the total order in quantity spaces were already discussed in section 5.2. Given that inequalities between points are valid, it is impossible to enforce the strict alternation between points and values in a quantity space. In order not to complicate the restrictions in points, the (in)equality relations used to specify the quantity space order are modelled in the interval class (see figure 11). The formalisation can be found in appendix B.10.3.

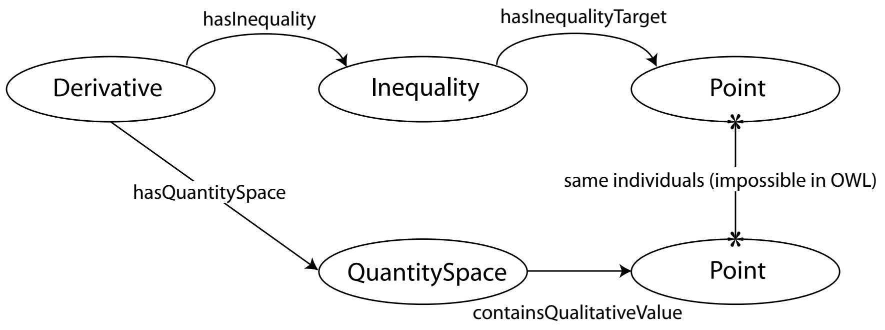

Figure 20: An (in)equality relations between a derivative and a point.

Interval Necessary Conditions:
$\forall\ hasInequality(Inequality \sqcap (hasInequalityTarget \geq 1) \sqcap$
$(\forall\ hasInequalityTarget\ Point))$

The hasValue Relation In order to facilitate easier modelling of (in)equality relations, Garp3 introduces hasValue relations. These relations are visualised as arrows which point at values of quantity spaces, indicating that the magnitude or derivative has that specific value. Unlike inequalities, hasValue relations may point to intervals. The qualitative reasoner translates hasValue relations with intervals to 2 inequalities. One stating that the magnitude (or derivative) has a value smaller than the point above the interval, and one stating the value is greater than the point below the interval. The formalisation is shown in appendix B.10.4.

Necessary hasValue relations:
$\forall\ hasValueTarget\ QualitativeValue$

6.2 Correspondences

Correspondences specify that values occur simultaneously. They normally occur between qualitative values of different quantity spaces, but it is also possible to create a correspondence relation between quantity spaces themselves. Those relations are an abbreviation for value correspondence relations between each of the quantity values of the different quantity spaces. Directed and undirected correspondences are distinguished. The former states that the target value may be derived from the origin value, while the latter allows the derivation of the value in both directions. Whether or not a correspondence is directed or not is indicated using the datatype property isDirected. Furthermore it is possible that quantity space correspondences are mirrored. This indicates that the first value of the first quantity space corresponds to the last one in the second quantity space, the last value in the first quantity space corresponds to the first one in the second quantity space, etc. This is indicated using the isMirrored datatype property.

The semantics of quantity space correspondences is unclear when the quantity spaces have a different amount of values, as it is unknown between which values value-correspondences would exist. The only way to formalise this in OWL is to create classes for each quantity space size, and restrict correspondences to members with the same number of qualitative values. The correct formalisation would take an infinite number of classes, therefore this restriction is not implemented. It would be beneficial if it would be possible to express in OWL that a domain and range should have the same cardinality. Another problem is that correspondences, just as configurations, should not be reflexive. It is a known problem that this restriction is inexpressible in OWL. For value correspondences a restriction is needed that restricts value correspondences to members of different quantity spaces. As OWL restrictions can only state that a range has to be of a specific type, this is impossible to implement. This leaves the basic restrictions that correspondences should be between members of the same class, and should have exactly one target, as is shown in figure 21. The formalisation in OWL is shown in appendix B.9.

Besides quantity space correspondences and value correspondences, there is a third kind of correspondence. The full correspondence indicates that both the magnitudes as the derivatives of two quantities have quantity space correspondences between them. The way this model ingredient will be represented in Garp3 is still pending, therefore no restrictions are added for full correspondences.

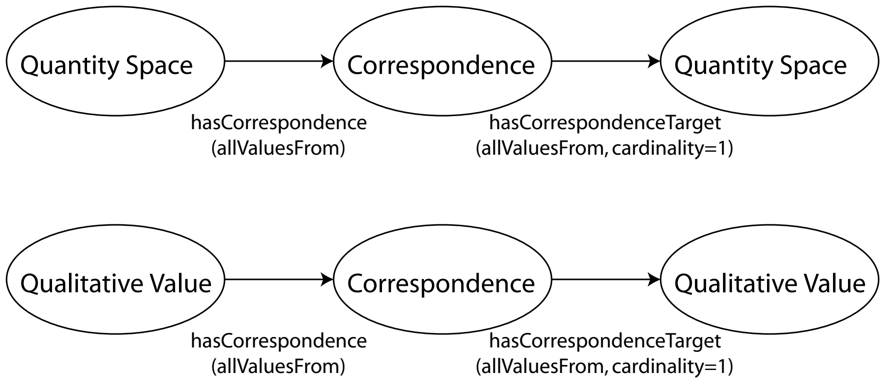
Quantity Space
Correspondence
Quantity Space
hasCorrespondence (allValuesFrom)
hasCorrespondenceTarget (allValuesFrom, cardinality=1)
Qualitative Value
Correspondence
Qualitative Value
hasCorrespondence (allValuesFrom)
hasCorrespondenceTarget (allValuesFrom, cardinality=1)

Figure 21: The only valid correspondence relations are between two quantity spaces, or two qualitative values.

QualitativeValue Restrictions:

$$\begin{aligned}&\forall\ hasCorrespondence(ValueCorrespondence \sqcap\\ &\quad(\forall\ hasCorrespondenceTarget\ QualitativeValue)\ \sqcap\\ &\quad(hasCausalCorrespondenceTarget\ =\ 1))\end{aligned}$$

Quantity Space Restrictions:

$$\begin{aligned}&\forall\ hasCorrespondence(QuantitySpaceCorrespondence \sqcap\\ &\quad(\forall\ hasCorrespondenceTarget\ QuantitySpace)\ \sqcap\\ &\quad(hasCausalCorrespondenceTarget\ =\ 1))\end{aligned}$$

6.3 Causal Dependencies

6.3.1 Proportionalities and Influences

Proportionalities and Influences are relations between quantities which indicate what quantities in a model change. These relations can be either positive or negative. A positive proportionality indicates that the derivative of the target quantity is positive if the derivative of the origin quantity is positive, and is negative if the derivative of the origin quantity is negative. For a negative proportionality this is just vice versa. A positive influence indicates that the target quantity derivative is positive if the magnitude of the origin quantity is greater than zero, and negative if it less than zero. For the negative influence this is just the other way around.

The semantics of the causal dependencies are modelled in the owner class, in this case the quantity class. Quantities can have influences and proportionalities as causal dependency relations, and they must have exactly one quantity as a target (see figure 22). The formalisation in OWL is shown in appendix B.8.1.

Necessary Quantity Restrictions

$\forall\ hasCausalDependency($
$\quad (Influence\ \sqcup\ Proportionality) \sqcap$
$\quad (\forall\ hasCausalDependencyTarget\ Quantity)\ \sqcap$
$\quad (hasCausalDependencyTarget\ =\ 1))$

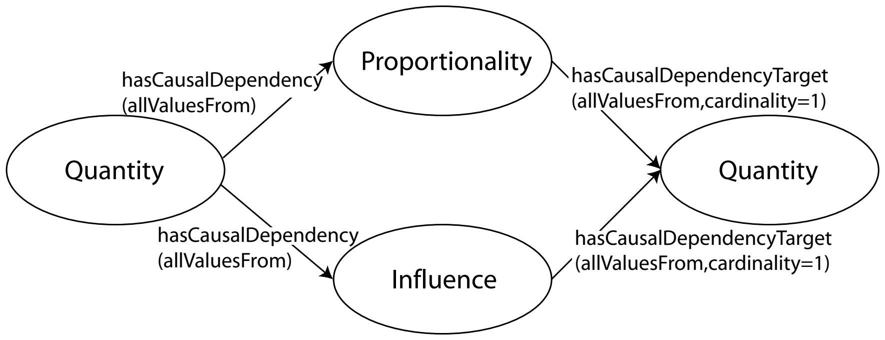
Proportionality
hasCausalDependency (allValuesFrom)
hasCausalDependencyTarget (allValuesFrom,cardinality=1)
Quantity
Quantity
hasCausalDependency (allValuesFrom)
hasCausalDependencyTarget (allValuesFrom,cardinality=1)
Influence

Figure 22: The use of proportionality and influences relations. Note that the restrictions applied to the hasCausalDependencyTarget relation are modelled in the Quantity class which owns the relation, and not in the influence or proportionality class.

Possible conditions	**Impossible conditions**
Entities	Correspondences
Configurations	Proportionalities
Agents	Influences
Attributes	
Quantities	
Inequalities	
Min/Plus	
Assumptions	
Model Fragments	
Possible consequences	**Impossible consequences**
Entities (except in static MF)	Agents
Configurations (except in static MF)	Assumptions
Attributes	Model Fragments
Quantities	
Inequalities	
Min/Plus	
Correspondences	
Proportionalities	
Influences	

Table 2: Condition and consequence abilities of QR ingredients.

7 Aggregates

Aggregates are model constituents consisting of multiple QR ingredients. *Model Fragments* and *Scenarios* are aggregate types. Model Fragments can be further subdivided into (1) *static fragments*, (2) *process fragments*, and (3) *agent fragments*.

7.1 Model Fragments

Model fragments consist of multiple model ingredients. As these ingredients are incorporated as either conditions or consequences, model fragments have the possible relations hasCondition and hasConsequence. Table 2 shows how certain model ingredients may be used as conditions and consequences. The formalisation of these restrictions in OWL can be found in appendix B.12.1.

Necessary Model Fragment Restrictions

$\forall\ hasCondition(Structural \sqcup Behavioural \sqcup$
$\quad AssumptionType \sqcup Inequality \sqcup PlusMin \sqcup ModelFragment)$

$\forall\ hasConsequence(Entity \sqcup Configuration \sqcup Attribute \sqcup$
$\quad AttributeValue \sqcup Behavioural \sqcup Dependency)$

7.1.1 Static Fragment

Static fragments are used to describe partial structures of systems. No influences causing change are formalised in static fragments, and thus no new entities, agents and configurations may be introduced (as consequences). Furthermore, no agents may be included as conditions. The formalisation in OWL can be found in appendix B.12.2.

Necessary & Sufficient Static Fragment Restrictions

$\neg(\exists\ hasCondition\ Agent)$

$\neg(\exists\ hasConsequence\ Configuration)$

$\neg(\exists\ hasConsequence\ Entity)$

$\neg(\exists\ hasConsequence\ Influence)$

$\neg(\exists\ hasConsequence\ Agent)$

7.1.2 Process Fragment

Process fragments are used to describe the processes which take place in a system. The processes in process fragments may not be caused by agent, which are therefore prohibited in these fragments. The formalisation in OWL is shown in appendix B.12.3.

Necessary & Sufficient Process Fragment Restrictions

$\neg(\exists\ hasCondition\ Agent)$

$\exists\ hasConsequence\ Influence$

7.1.3 Agent Fragment

Agent fragments describe situations in which an agent may interact with a system. Every model fragment which contains an agent should be an agent fragment. The formalisation in OWL can be found in appendix B.12.4.

Necessary & Sufficient Agent Fragment Restrictions

$\exists\ hasCondition\ Agent$

7.2 Scenarios

Scenarios describe situations of the modelled systems which becomes the start node of the state graph. All the ingredients which may be used as conditions in model fragments (table 2), except Model Fragments, may be used in scenarios. The OWL formalisation of the scenario concept can be found in appendix B.12.5. An OWL formalisation of an example scenario can be found in appendix C.8.

$$\forall\ hasFact(Structural\ \sqcup\ Behavioural\ \sqcup$$
$$AssumptionType\ \sqcup\ Inequality\ \sqcup\ PlusMin)$$

8 Special Model Ingredients

8.1 Assumptions

As it is likely that new types of assumptions will be introduced in the QR vocabulary, the AssumptionType concept is defined. For now, the only class which is part of this set is the generic assumption. Assumptions are neither structural nor behavioural ingredients, as they describe knowledge about the model, and not about the system. Assumption can optionally be related to an entity or agent, to indicate to which model ingredient the assumption refers. The OWL code of the assumption restrictions can be seen in appendix B.13.

Assumptions are used as conditions in model fragments to assert that something is true. They are usually used in combination with inequalities to reduce the possible behaviour of a system. Model builders can define their own assumptions in the same way they can define entities and agents. The formalisation of this subsumption hierarchy in OWL is equivalent to those of the entities and agents, as can be seen in appendix C.3.

Necessary Assumption restrictions:

$isAssumptionRegarding \ \leq \ 1$

$\forall \ isAssumptionRegarding \ (Entity \ \sqcup \ Agent)$

8.2 Identity relations

An identity indicates that two model ingredients are the same individual. The only reason to do this is when one of the model ingredients in incorporated via another model fragment. This makes it possible to indicate that two elements in different model fragments are actually the same.

9 Conclusions

In order to be able to share qualitative models among users and search for specific structures within them, the models have to be formalised in an open semantic format. This means an alternative format to represent Garp3 models was needed in addition to the current binary file format. We have presented the formalisation of the qualitative vocabulary in OWL, which functions as a sort of DTD for the qualitative models. A parser has been integrated in the Garp3 software which can import (and export) qualitative models from (to) OWL. The formalisation of these domain ontologies (models) has also been presented.

Using this result we will start working on the qualitative model repository and a search engine which can query for specific structures in QR models. Furthermore, we will index the repository using another OWL ontology.

References

[1] Sean Bechhofer, Frank van Harmelen, Jim Hendler, Ian Horrocks, Deborah L. McGuinness, Peter F. Patel-Schneider, and Lynn Andrea Stein. OWL web ontology language reference. W3C recommendation, World Wide Web Consortium (W3C), February 2004.

[2] Dave Beckett (editor) and Brian McBride (series editor). RDF/XML syntax specification (revised). W3C recommendation, World Wide Web Consortium (W3C), February 2004.

[3] Tim Bray, Dave Hollander, and Andrew Layman. Namespaces in XML. W3C recommendation, World Wide Web Consortium (W3C), January 1999.

[4] Tim Bray, Jean Paoli, C. M. Sperberg-McQueen, Eve Maler, and François Yergeau. Extensible markup language (XML) 1.0 (third edition). W3C recommendation, World Wide Web Consortium (W3C), February 2004.

[5] Dan Brickley (editor), R.V. Guha (editor), and Brian McBride (series editor). RDF vocabulary description language 1.0: RDF Schema. W3C recommendation, World Wide Web Consortium (W3C), February 2004.

[6] Kenneth D. Forbus. Qualitative process theory. *Artificial Intelligence*, 24(1-3):85–168, December 1984.

[7] Jan Grant (editor), Dave Beckett (series editor), and Brian McBride (series editor). RDF test cases. W3C recommendation, World Wide Web Consortium (W3C), February.

[8] Patrick Hayes (editor) and Brian McBride (series editor). RDF semantics. W3C recommendation, World Wide Web Consortium (W3C), February 2004.

[9] Gertjan van Heijst, Sabina Falasconi, Ameen Abu-Hanna, Guus Schreiber, and Mario Stefanelli. A case study in ontology library contruction. *Artificial Intelligence in Medicine*, 7(3):227–255, June 1995.

[10] Diane Hillmann. Using Dublin Core. DCMI recommended resource, Dublin Core Metadata Initiative, August 2003.

[11] Ian Horrocks, Peter F. Patel-Schneider, and Frank van Harmelen. From SHIQ and RDF to OWL: The making of a web ontology language. *Journal of Web Semantics*, 1(1):7–26, December 2003.

[12] Graham Klyne (editor), Jeremy J. Carroll (editor), and Brian McBride (series editor). Resource description framework (RDF): Concepts and abstract syntax. W3C recommendation, World Wide Web Consortium (W3C), February 2004.

[13] Hector J. Levesque and Ronald J. Brachman. *Readings in Knowledge Representation*, chapter A Fundamental Tradeoff in Knowledge Representation and Reasoning (Revised Version), pages 42–70. Morgan Kaufmann, 1985.

[14] Jochem Liem and Bert Bredeweg. OWL and qualitatative reasoning models. 2006. Submitted for publication.

[15] Frank Manola (editor), Eric Miller (editor), and Brian McBride (series editor). RDF primer. W3C recommendation, World Wide Web Consortium (W3C), 2004 2004.

[16] Natasha Noy and Alan Rector. Defining n-ary relations on the semantic web. W3C working draft, World Wide Web Consortium (W3C), January 2005.

[17] Peter F. Patel-Schneider, Patrick Hayes, and Ian Horrock. OWL web ontology language semantics and abstract syntax. W3C recommendation, World Wide Web Consortium (W3C), February 2004.

[18] Alan Rector. Representing specified values in OWL: "value partitions" and "value sets". W3C working draft, World Wide Web Consortium (W3C), August 2004.

[19] Michael K. Smith, Chris Welty, and Deborah L. McGuinness. OWL web ontology language guide. W3C recommendation, World Wide Web Consortium (W3C), February 2004.

A OWL Patterns

A.1 Property Values Patterns

A.1.1 Property Values as a Set of Individuals

```
<owl:Class rdf:ID="Meat" />
<owl:ObjectProperty rdf:ID="hasReadiness">
    <rdf:type rdf:resource="&owl;FunctionalProperty" />
    <rdfs:domain rdf:resource="#Meat" />
    <rdfs:range rdf:resource="#Readiness" />
</owl:ObjectProperty>
<owl:Class rdf:ID="Readiness">
    <owl:oneOf rdf:parseType="Collection">
        <owl:Thing rdf:about="#Rare" />
        <owl:Thing rdf:about="#MediumRare" />
        <owl:Thing rdf:about="#Medium" />
        <owl:Thing rdf:about="#WellDone" />
    </owl:oneOf>
</owl:Class>
<Readiness rdf:ID="Rare" />
<Readiness rdf:ID="MediumRare" />
<Readiness rdf:ID="Medium" />
<Readiness rdf:ID="WellDone" />
<owl:AllDifferent>
    <owl:distinctMembers rdf:parseType="Collection">
        <owl:Thing rdf:about="#Rare" />
        <owl:Thing rdf:about="#MediumRare" />
        <owl:Thing rdf:about="#Medium" />
        <owl:Thing rdf:about="#WellDone" />
    </owl:distinctMembers>
</owl:AllDifferent>
```

A.1.2 Property Values as a Set of Classes

```
<owl:Class rdf:ID="Meat" />
<owl:ObjectProperty rdf:ID="hasReadiness">
    <rdf:type rdf:resource="&owl;FunctionalProperty" />
    <rdfs:domain rdf:resource="#Meat" />
    <rdfs:range rdf:resource="#Readiness" />
</owl:ObjectProperty>
<owl:Class rdf:ID="Readiness">
    <owl:unionOf rdf:parseType="Collection">
        <owl:Class rdf:about="#Rare" />
        <owl:Class rdf:about="#MediumRare" />
        <owl:Class rdf:about="#Medium" />
        <owl:Class rdf:about="#WellDone" />
    </owl:unionOf>
</owl:Class>
<owl:Class rdf:ID="Rare">
    <rdfs:subClassOf rdf:resource="#Readiness" />
    <owl:disjointWith rdf:resource="#MediumRare" />
    <owl:disjointWith rdf:resource="#Medium" />
    <owl:disjointWith rdf:resource="#WellDone" />
</owl:Class>
<owl:Class rdf:ID="MediumRare">
    <rdfs:subClassOf rdf:resource="#Readiness" />
    <owl:disjointWith rdf:resource="#Rare" />
    <owl:disjointWith rdf:resource="#Medium" />
    <owl:disjointWith rdf:resource="#WellDone" />
</owl:Class>
<owl:Class rdf:ID="Medium">
    <rdfs:subClassOf rdf:resource="#Readiness" />
    <owl:disjointWith rdf:resource="#Rare" />
    <owl:disjointWith rdf:resource="#MediumRare" />
    <owl:disjointWith rdf:resource="#WellDone" />
</owl:Class>
<owl:Class rdf:ID="WellDone">
    <rdfs:subClassOf rdf:resource="#Readiness" />
    <owl:disjointWith rdf:resource="#Rare" />
    <owl:disjointWith rdf:resource="#MediumRare" />
    <owl:disjointWith rdf:resource="#Medium" />
</owl:Class>
<MediumRare rdf:ID="WellDone_1" />
<Meat rdf:ID="SirloinSteak">
    <hasReadiness rdf:resource="#WellDone_1" />
</Meat>
```

A.1.3 A List as a Property Value

```
<owl:Class rdf:ID="ArgumentList" />
<owl:ObjectProperty rdf:ID="RestOfList">
    <rdf:type rdf:resource="&owl;FunctionalProperty" />
    <rdfs:domain rdf:resource="#ArgumentList" />
    <rdfs:range rdf:resource="#ArgumentList" />
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="hasContents">
<rdf:type rdf:resource="&owl;FunctionalProperty" />
    <rdfs:domain rdf:resource="#ArgumentList" />
    <rdfs:range rdf:resource="&owl;Thing" />
</owl:ObjectProperty>
<ArgumentList rdf:ID="EmptyList" />
<owl:Class rdf:ID="MeatReadiness" />
<owl:Class rdf:ID="MeatReadinessValue" />
<owl:Class rdf:ID="Rare">
    <rdfs:subClassOf rdf:resource="#MeatReadiness" />
    <owl:disjointWith rdf:resource="#MediumRare" />
    <owl:disjointWith rdf:resource="#Medium" />
    <owl:disjointWith rdf:resource="#WellDone" />
</owl:Class>
<owl:Class rdf:ID="MediumRare">
    <rdfs:subClassOf rdf:resource="#MeatReadiness" />
    <owl:disjointWith rdf:resource="#Rare" />
    <owl:disjointWith rdf:resource="#Medium" />
    <owl:disjointWith rdf:resource="#WellDone" />
</owl:Class>
<owl:Class rdf:ID="Medium">
    <rdfs:subClassOf rdf:resource="#MeatReadiness" />
    <owl:disjointWith rdf:resource="#Rare" />
    <owl:disjointWith rdf:resource="#MediumRare" />
    <owl:disjointWith rdf:resource="#WellDone" />
</owl:Class>
<owl:Class rdf:ID="WellDone">
    <rdfs:subClassOf rdf:resource="#MeatReadiness" />
    <owl:disjointWith rdf:resource="#Rare" />
    <owl:disjointWith rdf:resource="#MediumRare" />
    <owl:disjointWith rdf:resource="#Medium" />
</owl:Class>
<Rare rdf:ID="Rare1"/>
<MediumRare rdf:ID="MediumRare1" />
<Medium rdf:ID="Medium1" />
<WellDone rdf:ID="WellDone1" />
<owl:ObjectProperty rdf:ID="hasReadinessSequence">
    <rdfs:domain rdf:resource="#MeatReadiness" />
    <rdfs:range  rdf:resource="#ArgumentList" />
</owl:ObjectProperty>
```

```
<MeatReadiness rdf:ID="SirloinSteakReadiness">
    <hasReadinessSequence>
        <ArgumentList rdf:ID="List1">
            <hasContents rdf:resource="#Rare1" />
            <RestOfList>
                <ArgumentList rdf:ID="List2">
                    <hasContents rdf:resource="#MediumRare1" />
                    <RestOfList>
                        <ArgumentList rdf:ID="List3">
                            <hasContents rdf:resource="#Medium1" />
                            <RestOfList>
                                <ArgumentList rdf:ID="List4">
                                    <hasContents rdf:resource="#WellDone1" />
                                    <RestOfList rdf:resource="#EmptyList" />
                                </ArgumentList>
                            </RestOfList>
                        </ArgumentList>
                    </RestOfList>
                </ArgumentList>
            </RestOfList>
        </ArgumentList>
    </hasReadinessSequence>
</MeatReadiness>
```

B Qualitative Reasoning Vocabulary

B.1 The Qualitative Vocabulary Hierarchy

```
<owl:Class rdf:ID="QualitativeModelIngredient">
    <rdfs:label xml:lang="en">Qualitative Model Ingredient</rdfs:label>
    <rdfs:comment xml:lang="en">
        Qualitative Model Ingredients are either Building Blocks or
        Aggregates.
    </rdfs:comment>
    <owl:unionOf rdf:parseType="Collection">
        <owl:Class rdf:about="#BuildingBlock" />
        <owl:Class rdf:about="#Aggregate" />
    </owl:unionOf>
</owl:Class>
<owl:Class rdf:ID="BuildingBlock">
    <rdfs:label xml:lang="en">Building Block</rdfs:label>
    <rdfs:comment xml:lang="en">
        Building Blocks are single model ingredients.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#QualitativeModelIngredient" />
    <owl:disjointWith rdf:resource="#Aggregate" />
</owl:Class>
<owl:Class rdf:ID="Aggregate">
    <rdfs:label xml:lang="en">Aggregate</rdfs:label>
    <rdfs:comment xml:lang="en">
        Aggregates are composed out of multiple Building Blocks.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#QualitativeModelIngredient" />
    <owl:disjointWith rdf:resource="#BuildingBlock" />
</owl:Class>
<owl:Class rdf:ID="Structural">
    <rdfs:label xml:lang="en">Structural</rdfs:label>
    <rdfs:comment xml:lang="en">
        Structurals describe the structure of a modeled system.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#BuildingBlock" />
    <owl:disjointWith rdf:resource="#AssumptionType" />
    <owl:disjointWith rdf:resource="#Behavioural" />
</owl:Class>
<owl:Class rdf:ID="Identity">
    <rdfs:label xml:lang="en">Identity</rdfs:label>
    <rdfs:comment xml:lang="en">
        An identity indicates that two model ingredients are the same individual.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#BuildingBlock" />
    <owl:disjointWith rdf:resource="#Structural" />
    <owl:disjointWith rdf:resource="#Behavioural" />
    <owl:disjointWith rdf:resource="#AssumptionType" />
</owl:Class>
<owl:Class rdf:ID="AssumptionType">
    <rdfs:label xml:lang="en">Assumption Type</rdfs:label>
    <rdfs:comment xml:lang="en">
        Assumption Type is the set of possible Assumptions types.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#BuildingBlock" />
    <owl:disjointWith rdf:resource="#Structural" />
    <owl:disjointWith rdf:resource="#Behavioural" />
</owl:Class>
<owl:Class rdf:ID="Behavioural">
    <rdfs:label xml:lang="en">Behavioural</rdfs:label>
    <rdfs:comment xml:lang="en">
        Behavioural ingredients describe the behavioural aspects of a
        modeled system.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#BuildingBlock" />
    <owl:disjointWith rdf:resource="#AssumptionType" />
    <owl:disjointWith rdf:resource="#Structural" />
</owl:Class>
<owl:Class rdf:ID="Dependency">
    <rdfs:label xml:lang="en">Dependency</rdfs:label>
    <rdfs:comment xml:lang="en">
        Dependencies describe the behavioural relations between behavioural
        ingredients.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Behavioural" />
    <owl:disjointWith rdf:resource="#Derivative" />
    <owl:disjointWith rdf:resource="#Magnitude" />
    <owl:disjointWith rdf:resource="#QualitativeValue" />
    <owl:disjointWith rdf:resource="#Quantity" />
    <owl:disjointWith rdf:resource="#QuantitySpace" />
</owl:Class>
<!-- Mathematics: A number assigned to a quantity so that it may be compared
to other quantities. -->
<owl:Class rdf:ID="Mathematical">
    <rdfs:label xml:lang="en">Mathematical</rdfs:label>
    <rdfs:comment xml:lang="en">
        Mathematical dependencies describe the mathematical relations
        between behavioural ingredients.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Dependency" />
    <owl:disjointWith rdf:resource="#Correspondence" />
    <owl:disjointWith rdf:resource="#CausalDependency" />
</owl:Class>
<owl:Class rdf:ID="Entity">
    <rdfs:label xml:lang="en">Entity</rdfs:label>
    <rdfs:comment xml:lang="en">
        Entities are the structural ingredients a system is composed of.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Structural" />
    <owl:disjointWith rdf:resource="#Agent" />
    <owl:disjointWith rdf:resource="#Configuration" />
    <owl:disjointWith rdf:resource="#Attribute" />
    <owl:disjointWith rdf:resource="#AttributeValue" />
</owl:Class>
<owl:Class rdf:ID="Agent">
    <rdfs:label xml:lang="en">Agent</rdfs:label>
    <rdfs:comment xml:lang="en">
        Agents are not part of the inherent structure of the system, but can
        influence the system.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Structural" />
    <owl:disjointWith rdf:resource="#Entity" />
    <owl:disjointWith rdf:resource="#Configuration" />
    <owl:disjointWith rdf:resource="#Attribute" />
    <owl:disjointWith rdf:resource="#AttributeValue" />
</owl:Class>
<owl:Class rdf:ID="Configuration">
    <rdfs:label xml:lang="en">Configuration</rdfs:label>
    <rdfs:comment xml:lang="en">
        Configurations describe the relations between structural
        ingredients.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Structural" />
    <owl:disjointWith rdf:resource="#Agent" />
    <owl:disjointWith rdf:resource="#Entity" />
    <owl:disjointWith rdf:resource="#Attribute" />
    <owl:disjointWith rdf:resource="#AttributeValue" />
</owl:Class>
<owl:Class rdf:ID="Attribute">
    <rdfs:label xml:lang="en">Attribute</rdfs:label>
    <rdfs:comment xml:lang="en">
        Attributes describe the properties of entities and agents.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Structural" />
    <owl:disjointWith rdf:resource="#Agent" />
    <owl:disjointWith rdf:resource="#Configuration" />
    <owl:disjointWith rdf:resource="#Entity" />
    <owl:disjointWith rdf:resource="#AttributeValue" />
</owl:Class>
<owl:Class rdf:ID="AttributeValue">
    <rdfs:label xml:lang="en">Attribute Value</rdfs:label>
    <rdfs:comment xml:lang="en">
        AttributeValues are the possible values of attributes.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Structural" />
    <owl:disjointWith rdf:resource="#Agent" />
    <owl:disjointWith rdf:resource="#Configuration" />
    <owl:disjointWith rdf:resource="#Attribute" />
    <owl:disjointWith rdf:resource="#Entity" />
</owl:Class>
<owl:Class rdf:ID="Quantity">
    <rdfs:label xml:lang="en">Quantity</rdfs:label>
    <rdfs:comment xml:lang="en">
        Quantities are the changable features of an entity of agent.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Behavioural" />
    <owl:disjointWith rdf:resource="#Magnitude" />
    <owl:disjointWith rdf:resource="#Derivative" />
    <owl:disjointWith rdf:resource="#QuantitySpace" />
    <owl:disjointWith rdf:resource="#QualitativeValue" />
    <owl:disjointWith rdf:resource="#Dependency" />
</owl:Class>
<owl:Class rdf:ID="Assumption">
```

```
    <rdfs:label xml:lang="en">Assumption</rdfs:label>
    <rdfs:comment xml:lang="en">
        An assumption is an assertion that something is true.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#AssumptionType" />
</owl:Class>
<owl:Class rdf:ID="Magnitude">
    <rdfs:label xml:lang="en">Magnitude</rdfs:label>
    <rdfs:comment xml:lang="en">
        The magnitude is the zero-derivative of a quantity. A value can be
        assigned to it so it can be compared to other quantities.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Behavioural" />
    <owl:disjointWith rdf:resource="#Quantity" />
    <owl:disjointWith rdf:resource="#Derivative" />
    <owl:disjointWith rdf:resource="#QuantitySpace" />
    <owl:disjointWith rdf:resource="#QualitativeValue" />
    <owl:disjointWith rdf:resource="#Dependency" />
</owl:Class>
<owl:Class rdf:ID="Derivative">
    <rdfs:label xml:lang="en">Derivative</rdfs:label>
    <rdfs:comment xml:lang="en">
        The derivative is the first derivative of a quantity, indicating if
        the quantity is increasing, decreasing or stable. A value can be
        assigned to it so it can be compared to other quantities.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Behavioural" />
    <owl:disjointWith rdf:resource="#Magnitude" />
    <owl:disjointWith rdf:resource="#Quantity" />
    <owl:disjointWith rdf:resource="#QuantitySpace" />
    <owl:disjointWith rdf:resource="#QualitativeValue" />
    <owl:disjointWith rdf:resource="#Dependency" />
</owl:Class>
<owl:Class rdf:ID="QuantitySpace">
    <rdfs:label xml:lang="en">Quantity Space</rdfs:label>
    <rdfs:comment xml:lang="en">
        A quantity space is a list of possible values the magnitude or
        derivative of a quantity can become.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Behavioural" />
    <owl:disjointWith rdf:resource="#Magnitude" />
    <owl:disjointWith rdf:resource="#Derivative" />
    <owl:disjointWith rdf:resource="#Quantity" />
    <owl:disjointWith rdf:resource="#QualitativeValue" />
    <owl:disjointWith rdf:resource="#Dependency" />
</owl:Class>
<owl:Class rdf:ID="QualitativeValue">
    <rdfs:label xml:lang="en">Qualitative Value</rdfs:label>
    <rdfs:comment xml:lang="en">
        A qualitative value is either a point or interval which can become
        the value of the magnitude or derivative of a quantity.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Behavioural" />
    <owl:disjointWith rdf:resource="#Magnitude" />
    <owl:disjointWith rdf:resource="#Derivative" />
    <owl:disjointWith rdf:resource="#Quantity" />
    <owl:disjointWith rdf:resource="#QuantitySpace" />
    <owl:disjointWith rdf:resource="#Dependency" />
</owl:Class>
<owl:Class rdf:ID="Point">
    <rdfs:label xml:lang="en">Point</rdfs:label>
    <rdfs:comment xml:lang="en">
        A point is a qualitative value.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#QualitativeValue" />
    <owl:disjointWith rdf:resource="#Interval" />
</owl:Class>
<Point rdf:ID="Zero">
    <rdfs:label>Zero</rdfs:label>
</Point>
<owl:Class rdf:ID="Interval">
    <rdfs:label xml:lang="en">Point</rdfs:label>
    <rdfs:comment xml:lang="en">
        An interval is a qualitative value.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#QualitativeValue" />
    <owl:disjointWith rdf:resource="#Point" />
</owl:Class>
<owl:Class rdf:ID="CausalDependency">
    <rdfs:label xml:lang="en">Causal Dependency</rdfs:label>
    <rdfs:comment xml:lang="en">
        Causal dependencies are dependencies indicate which quantities are
        proportional, and which ones influence each other.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Dependency" />
    <owl:disjointWith rdf:resource="#Correspondence" />
    <owl:disjointWith rdf:resource="#Mathematical" />
</owl:Class>
<owl:Class rdf:ID="Proportionality">
    <rdfs:label xml:lang="en">Proportionality</rdfs:label>
    <rdfs:comment xml:lang="en">
        Proportionalities indicate which quantity changes if the other
        changes.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#CausalDependency" />
    <owl:disjointWith rdf:resource="#Influence" />
</owl:Class>
<owl:Class rdf:ID="PositiveProportionality">
    <rdfs:label xml:lang="en">Positive Proportionality</rdfs:label>
    <rdfs:comment xml:lang="en">
        A positive proportionality indicates that the target quantity
        increases if the origin quantity increases, and decreases if the
        origin quantity descreases.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Proportionality" />
    <owl:disjointWith rdf:resource="#NegativeProportionality" />
</owl:Class>
<owl:Class rdf:ID="NegativeProportionality">
    <rdfs:label xml:lang="en">Negative Proportionality</rdfs:label>
    <rdfs:comment xml:lang="en">
        A negative proportionality indicates that the target quantity
        decreases if the origin quantity increases, and increases if the
        origin quantity decreases.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Proportionality" />
    <owl:disjointWith rdf:resource="#PositiveProportionality" />
</owl:Class>
<owl:Class rdf:ID="Influence">
    <rdfs:label xml:lang="en">Influence</rdfs:label>
    <rdfs:comment xml:lang="en">
        Influences indicate that the target quantity changes if magnitude of
        the origin quantity is not zero.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#CausalDependency" />
    <owl:disjointWith rdf:resource="#Proportionality" />
</owl:Class>
```

```
<owl:Class rdf:ID="PositiveInfluence">
    <rdfs:label xml:lang="en">Positive Influence</rdfs:label>
    <rdfs:comment xml:lang="en">
        A positive influence indicates that the target quantity increases if
        the magnitude of the origin quantity is greater than zero, and
        decreases if it less than zero.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Influence" />
    <owl:disjointWith rdf:resource="#NegativeInfluence" />
</owl:Class>
<owl:Class rdf:ID="NegativeInfluence">
    <rdfs:label xml:lang="en">Negative Influence</rdfs:label>
    <rdfs:comment xml:lang="en">
        A negative influence indicates that the target quantity decreases if
        the magnitude of the origin quantity is greater than zero, and
        increases if it less than zero.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Influence" />
    <owl:disjointWith rdf:resource="#PositiveInfluence" />
</owl:Class>
<owl:Class rdf:ID="Correspondence">
    <rdfs:label xml:lang="en">Correspondence</rdfs:label>
    <rdfs:comment xml:lang="en">
        Correspondences specify that values occur simultaniously.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Dependency" />
    <owl:disjointWith rdf:resource="#Mathematical" />
    <owl:disjointWith rdf:resource="#CausalDependency" />
</owl:Class>
  <owl:Class rdf:ID="Correspondence">
        <rdfs:label xml:lang="en">Correspondence</rdfs:label>
        <rdfs:comment xml:lang="en">
            Correspondences specify that values occur simultaniously.
        </rdfs:comment>
        <rdfs:subClassOf rdf:resource="#Dependency" />
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="&qr;isDirected" />
            <owl:maxCardinality rdf:datatype="&xsd;nonNegativeInteger"
            >1</owl:maxCardinality>
        </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <owl:Restriction>
        <owl:onProperty rdf:resource="&qr;isInverted" />
            <owl:maxCardinality rdf:datatype="&xsd;nonNegativeInteger"
            >1</owl:maxCardinality>
        </owl:Restriction>
    </rdfs:subClassOf>
    <owl:disjointWith rdf:resource="#Mathematical" />
    <owl:disjointWith rdf:resource="#CausalDependency" />
</owl:Class>
<owl:DatatypeProperty rdf:ID="isDirected" />
<owl:DatatypeProperty rdf:ID="isInverted" />
<owl:Class rdf:ID="ValueCorrespondence">
    <rdfs:label xml:lang="en">Value Correspondence</rdfs:label>
    <rdfs:comment xml:lang="en">
        A Value Correspondence specifies that two values occur simultaniously.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Correspondence" />
    <owl:disjointWith rdf:resource="#QuantitySpaceCorrespondence" />
    <owl:disjointWith rdf:resource="#FullCorrespondence" />
</owl:Class>
```

```
<owl:Class rdf:ID="QuantitySpaceCorrespondence">
    <rdfs:label xml:lang="en">Quantity Space Correspondence</rdfs:label>
    <rdfs:comment xml:lang="en">
    Quantity Space Correspondences are used to denote that the values of two quantity spaces
    correspond to each other.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Correspondence" />
    <owl:disjointWith rdf:resource="#ValueCorrespondence" />
    <owl:disjointWith rdf:resource="#FullCorrespondence" />
</owl:Class>
<owl:Class rdf:ID="FullCorrespondence">
    <rdfs:label xml:lang="en">Full Correspondence</rdfs:label>
    <rdfs:comment xml:lang="en">
        Full Correspondences indicate that both the quantity spaces of the magnitudes as the quantity spaces of the
        derivatives correspond to eachother.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Correspondence" />
    <owl:disjointWith rdf:resource="#ValueCorrespondence" />
    <owl:disjointWith rdf:resource="#QuantitySpaceCorrespondence" />
</owl:Class>
<owl:Class rdf:ID="Inequality">
    <rdfs:label xml:lang="en">Inequality</rdfs:label>
    <rdfs:comment xml:lang="en">
    Inequalities are used to indicate the difference or equality between values.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Mathematical" />
    <owl:disjointWith rdf:resource="#Calculus" />
</owl:Class>
<owl:Class rdf:ID="SmallerThan">
    <rdfs:label xml:lang="en">Smaller Than</rdfs:label>
    <rdfs:comment xml:lang="en">
    A smaller than relation indicates that the origin value is smaller than the target value.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Inequality" />
    <owl:disjointWith rdf:resource="#SmallerOrEqualTo" />
    <owl:disjointWith rdf:resource="#EqualTo" />
    <owl:disjointWith rdf:resource="#GreaterOrEqualTo" />
    <owl:disjointWith rdf:resource="#GreaterThan" />
    <owl:disjointWith rdf:resource="#hasValue" />
</owl:Class>
<owl:Class rdf:ID="SmallerOrEqualTo">
    <rdfs:label xml:lang="en">Smaller Or Equal To</rdfs:label>
    <rdfs:comment xml:lang="en">
    A smaller or equal to relation indicates that the origin value is smaller or equal to the target value.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Inequality" />
    <owl:disjointWith rdf:resource="#SmallerThan" />
    <owl:disjointWith rdf:resource="#EqualTo" />
    <owl:disjointWith rdf:resource="#GreaterOrEqualTo" />
    <owl:disjointWith rdf:resource="#GreaterThan" />
    <owl:disjointWith rdf:resource="#hasValue" />
</owl:Class>
<owl:Class rdf:ID="EqualTo">
    <rdfs:label xml:lang="en">Equal To</rdfs:label>
    <rdfs:comment xml:lang="en">
    An equality relation indicates that the origin value is equal to the target value.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Inequality" />
    <owl:disjointWith rdf:resource="#SmallerThan" />
    <owl:disjointWith rdf:resource="#SmallerOrEqualTo" />
    <owl:disjointWith rdf:resource="#GreaterOrEqualTo" />
    <owl:disjointWith rdf:resource="#GreaterThan" />
    <owl:disjointWith rdf:resource="#hasValue" />
```

```
</owl:Class>
<owl:Class rdf:ID="GreaterOrEqualTo">
    <rdfs:label xml:lang="en">Greater Or Equal To</rdfs:label>
    <rdfs:comment xml:lang="en">
    A greater or equal to relation indicates that the origin value is greater or equal to the target value.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Inequality" />
    <owl:disjointWith rdf:resource="#SmallerThan" />
    <owl:disjointWith rdf:resource="#SmallerOrEqualTo" />
    <owl:disjointWith rdf:resource="#EqualTo" />
    <owl:disjointWith rdf:resource="#GreaterThan" />
    <owl:disjointWith rdf:resource="#hasValue" />
</owl:Class>
<owl:Class rdf:ID="GreaterThan">
    <rdfs:label xml:lang="en">Greater Than</rdfs:label>
    <rdfs:comment xml:lang="en">
    A greater than relation indicates that the origin value is greater than the target value.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Inequality" />
    <owl:disjointWith rdf:resource="#SmallerThan" />
    <owl:disjointWith rdf:resource="#SmallerOrEqualTo" />
    <owl:disjointWith rdf:resource="#EqualTo" />
    <owl:disjointWith rdf:resource="#GreaterOrEqualTo" />
    <owl:disjointWith rdf:resource="#hasValue" />
</owl:Class>
<owl:Class rdf:ID="hasValue">
    <rdfs:label xml:lang="en">hasValue</rdfs:label>
    <rdfs:comment xml:lang="en">
        hasValue relations indicate a magnitude or derivative have a certain value.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Inequality" />
    <owl:disjointWith rdf:resource="#SmallerThan" />
    <owl:disjointWith rdf:resource="#SmallerOrEqualTo" />
    <owl:disjointWith rdf:resource="#EqualTo" />
    <owl:disjointWith rdf:resource="#GreaterOrEqualTo" />
    <owl:disjointWith rdf:resource="#GreaterThan" />
</owl:Class>
<owl:Class rdf:ID="Calculus">
    <rdfs:label xml:lang="en">Plus/Min</rdfs:label>
    <rdfs:comment xml:lang="en">
    Calculus is the set of addition and substraction relations. The relation
    itself can have multiple inequality relations.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Mathematical" />
    <owl:disjointWith rdf:resource="#Inequality" />
</owl:Class>
<owl:Class rdf:ID="Plus">
    <rdfs:label xml:lang="en">Plus</rdfs:label>
    <rdfs:comment xml:lang="en">
    Plus is addition of the origin and target value and can have multiple
    inequalities.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Calculus" />
    <owl:disjointWith rdf:resource="#Min" />
</owl:Class>
<owl:Class rdf:ID="Min">
    <rdfs:label xml:lang="en">Plus/Min</rdfs:label>
    <rdfs:comment xml:lang="en">
    Min is substraction of the origin and target value and can have multiple
    inequalities.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Calculus" />
    <owl:disjointWith rdf:resource="#Plus" />
```

```
</owl:Class>
<owl:Class rdf:ID="ModelFragment">
    <rdfs:label xml:lang="en">Model Fragment</rdfs:label>
    <rdfs:comment xml:lang="en">
    A model fragment describe parts of the structure and behaviour of a
    system and is composed of multiple building blocks.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Aggregate" />
    <owl:disjointWith rdf:resource="#Scenario" />
</owl:Class>
<owl:Class rdf:ID="StaticFragment">
    <rdfs:label xml:lang="en">Static Fragment</rdfs:label>
    <rdfs:comment xml:lang="en">
    A static fragment is a model fragment without influences or agents.
    </rdfs:comment>
    <owl:disjointWith rdf:resource="#ProcessFragment" />
    <owl:disjointWith rdf:resource="#AgentFragment" />
</owl:Class>
<owl:Class rdf:ID="ProcessFragment">
    <rdfs:label xml:lang="en">Process Fragment</rdfs:label>
    <rdfs:comment xml:lang="en">
    A process fragment is a model fragment which models a process, but does
    not contain agents.
    </rdfs:comment>
    <owl:disjointWith rdf:resource="#StaticFragment" />
    <owl:disjointWith rdf:resource="#AgentFragment" />
</owl:Class>
<owl:Class rdf:ID="AgentFragment">
    <rdfs:label xml:lang="en">Agent Fragment</rdfs:label>
    <rdfs:comment xml:lang="en">
    An agent fragment is a model fragment which contains an agent.
    </rdfs:comment>
    <owl:disjointWith rdf:resource="#ProcessFragment" />
    <owl:disjointWith rdf:resource="#StaticFragment" />
</owl:Class>
<owl:Class rdf:ID="Scenario">
    <rdfs:label xml:lang="en">Scenario</rdfs:label>
    <rdfs:comment xml:lang="en">
    A scenario describes a situation of the system which becomes the
    start node of the state graph.
    </rdfs:comment>
    <rdfs:subClassOf rdf:resource="#Aggregate" />
    <owl:disjointWith rdf:resource="#ModelFragment" />
</owl:Class>
```

B.2 Qualitative Model Ingredient Restrictions

```
<owl:Class rdf:about="#QualitativeModelIngredient">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#has_xposition_on_screen" />
            <owl:maxCardinality rdf:datatype="&xsd;nonNegativeInteger"
            >1</owl:maxCardinality>
        </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#has_yposition_on_screen" />
            <owl:maxCardinality rdf:datatype="&xsd;nonNegativeInteger"
            >1</owl:maxCardinality>
        </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
```

```
        <owl:Restriction>
            <owl:onProperty rdf:resource="#is_hidden_on_screen" />
            <owl:maxCardinality rdf:datatype="&xsd;nonNegativeInteger"
            >1</owl:maxCardinality>
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
<owl:DatatypeProperty rdf:ID="has_xposition_on_screen" />
<owl:DatatypeProperty rdf:ID="has_yposition_on_screen" />
<owl:DatatypeProperty rdf:ID="is_hidden_on_screen" />
```

B.3 Entity and Agent Restrictions

```
<owl:Class rdf:about="#Entity">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasAttribute" />
            <owl:allValuesFrom rdf:resource="#Attribute" />
        </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasQuantity" />
            <owl:allValuesFrom rdf:resource="#Quantity" />
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="#Agent">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasAttribute" />
            <owl:allValuesFrom rdf:resource="#Attribute" />
        </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasQuantity" />
            <owl:allValuesFrom rdf:resource="#Quantity" />
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
<owl:ObjectProperty rdf:ID="hasConfiguration" />
<owl:ObjectProperty rdf:ID="hasAttribute" />
<owl:ObjectProperty rdf:ID="hasQuantity" />
```

B.4 Configuration Restrictions

```
<owl:Class rdf:about="#Configuration">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasConfigurationTarget" />
            <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">1
            </owl:cardinality>
        </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasConfigurationOwner" />
            <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">1
            </owl:cardinality>
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
```

```
<owl:ObjectProperty rdf:ID="hasConfigurationTarget" />
<owl:ObjectProperty rdf:ID="isConfigurationTargetOf">
    <owl:inverseOf rdf:resource="#hasConfigurationTarget" />
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="hasConfigurationOwner">
    <owl:inverseOf rdf:resource="#hasConfiguration" />
</owl:ObjectProperty>
```

B.4.1 Entity and Agent Configuration Restrictions

```
<owl:Class rdf:about="#Agent">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasConfiguration" />
            <owl:allValuesFrom>
                <owl:Class>
                    <owl:intersectionOf rdf:parseType="Collection">
                        <owl:Class rdf:about="#Configuration" />
                        <owl:Restriction>
                            <owl:onProperty rdf:resource="#hasConfigurationTarget" />
                            <owl:someValuesFrom>
                                <owl:Class>
                                    <owl:unionOf rdf:parseType="Collection">
                                        <owl:Class rdf:about="#Entity" />
                                        <owl:Class rdf:about="#Agent" />
                                    </owl:unionOf>
                                </owl:Class>
                            </owl:someValuesFrom>
                        </owl:Restriction>
                    </owl:intersectionOf>
                </owl:Class>
            </owl:allValuesFrom>
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="#Entity">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasConfiguration" />
            <owl:allValuesFrom>
                <owl:Class>
                    <owl:intersectionOf rdf:parseType="Collection">
                        <owl:Class rdf:about="#Configuration" />
                        <owl:Restriction>
                            <owl:onProperty rdf:resource="#hasConfigurationTarget" />
                            <owl:someValuesFrom>
                                <owl:Class>
                                    <owl:unionOf rdf:parseType="Collection">
                                        <owl:Class rdf:about="#Entity" />
                                        <owl:Class rdf:about="#Agent" />
                                    </owl:unionOf>
                                </owl:Class>
                            </owl:someValuesFrom>
                        </owl:Restriction>
                    </owl:intersectionOf>
                </owl:Class>
            </owl:allValuesFrom>
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
```

B.5 Attribute Restrictions

```
<owl:Class rdf:about="#Attribute">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasAttributeValue" />
            <owl:allValuesFrom rdf:resource="#AttributeValue" />
        </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasAttributeValue" />
            <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">1
            </owl:cardinality>
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
<owl:ObjectProperty rdf:ID="hasAttributeValue" />
```

B.6 Quantity Restrictions

```
<owl:Class rdf:about="#Quantity">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasMagnitude" />
            <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">1
            </owl:cardinality>
        </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasMagnitude" />
            <owl:allValuesFrom rdf:resource="#Magnitude" />
        </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasDerivative" />
            <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">1
            </owl:cardinality>
        </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasDerivative" />
            <owl:allValuesFrom rdf:resource="#Derivative" />
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="#Magnitude">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasQuantitySpace" />
            <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">1
            </owl:cardinality>
        </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasQuantitySpace" />
            <owl:allValuesFrom rdf:resource="#QuantitySpace" />
        </owl:Restriction>
    </rdfs:subClassOf>
```

```
</owl:Class>
<owl:Class rdf:about="#Derivative">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasQuantitySpace" />
            <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">1
            </owl:cardinality>
        </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasQuantitySpace" />
            <owl:allValuesFrom rdf:resource="#QuantitySpace" />
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
<owl:ObjectProperty rdf:ID="hasMagnitude">
    <rdfs:range  rdf:resource="#Magnitude" />
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="magnitudeBelongsToQuantity">
    <owl:inverseOf rdf:resource="#hasMagnitude" />
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="hasDerivative">
    <rdfs:range  rdf:resource="#Derivative" />
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="derivativeBelongsToQuantity">
    <owl:inverseOf rdf:resource="#hasDerivative" />
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="hasQuantitySpace">
    <rdfs:range  rdf:resource="#QuantitySpace" />
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="isQuantitySpaceOf">
    <owl:inverseOf rdf:resource="#hasQuantitySpace" />
    <rdfs:domain rdf:resource="#QuantitySpace" />
</owl:ObjectProperty>
```

B.7 Quantity Space Restrictions

```
<owl:Class rdf:about="#QuantitySpace">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#containsQualitativeValue" />
            <owl:minCardinality rdf:datatype="&xsd;nonNegativeInteger">1
            </owl:minCardinality>
        </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#containsQualitativeValue" />
            <owl:allValuesFrom rdf:resource="#QualitativeValue" />
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
<owl:ObjectProperty rdf:ID="containsQualitativeValue">
    <rdfs:range  rdf:resource="#QualitativeValue" />
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="belongsToQuantitySpace">
    <owl:inverseOf rdf:resource="#containsQualitativeValue" />
</owl:ObjectProperty>
```

B.8 Causal Dependency Restrictions

B.8.1 Quantity Causal Dependency Restrictions

```
<owl:Class rdf:about="#Quantity">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasCausalDependency" />
            <owl:allValuesFrom>
                <owl:Class>
                    <owl:intersectionOf rdf:parseType="Collection">
                        <owl:Class>
                            <owl:unionOf rdf:parseType="Collection">
                                <owl:Class rdf:about="#Influence" />
                                <owl:Class rdf:about="#Proportionality" />
                            </owl:unionOf>
                        </owl:Class>
                        <owl:Restriction>
                            <owl:onProperty rdf:resource="#hasCausalDependencyTarget" />
                            <owl:allValuesFrom rdf:resource="#Quantity" />
                        </owl:Restriction>
                        <owl:Restriction>
                            <owl:onProperty rdf:resource="#hasCausalDependencyTarget" />
                            <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">1
                            </owl:cardinality>
                        </owl:Restriction>
                    </owl:intersectionOf>
                </owl:Class>
            </owl:allValuesFrom>
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
<owl:ObjectProperty rdf:ID="hasCausalDependency" />
<owl:ObjectProperty rdf:ID="hasCausalDependencyTarget" />
```

B.9 Correspondences

```
<owl:ObjectProperty rdf:ID="hasCorrespondence" />
<owl:ObjectProperty rdf:ID="hasCorrespondenceTarget" />
<owl:Class rdf:about="#QuantitySpace">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasCorrespondence" />
            <owl:allValuesFrom>
                <owl:Class>
                    <owl:intersectionOf rdf:parseType="Collection">
                        <owl:Class rdf:about="#QuantitySpaceCorrespondence" />
                        <owl:Restriction>
                            <owl:onProperty rdf:resource="#hasCorrespondenceTarget" />
                            <owl:allValuesFrom rdf:resource="#QuantitySpace" />
                        </owl:Restriction>
                        <owl:Restriction>
                            <owl:onProperty rdf:resource="#hasCorrespondenceTarget" />
                            <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">1
                            </owl:cardinality>
                        </owl:Restriction>
                    </owl:intersectionOf>
                </owl:Class>
            </owl:allValuesFrom>
            <!-- The domain and range should have the same cardinality, but this
            is not expressable in OWL -->
        </owl:Restriction>
    </rdfs:subClassOf>
```

```
</owl:Class>
<owl:Class rdf:about="#QualitativeValue">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasCorrespondence" />
            <owl:allValuesFrom>
                <owl:Class>
                    <owl:intersectionOf rdf:parseType="Collection">
                        <owl:Class rdf:about="#ValueCorrespondence" />
                        <owl:Restriction>
                            <owl:onProperty rdf:resource="#hasCorrespondenceTarget" />
                            <owl:allValuesFrom rdf:resource="#QualitativeValue" />
                        </owl:Restriction>
                        <owl:Restriction>
                            <owl:onProperty rdf:resource="#hasCorrespondenceTarget" />
                            <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">1
                            </owl:cardinality>
                        </owl:Restriction>
                    </owl:intersectionOf>
                </owl:Class>
            </owl:allValuesFrom>
            <!-- The qualitative values should not be in the same
            quantity space, but this is not expressable in OWL -->
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
```

B.10 Inequalities

B.10.1 Inequalities Belonging to Points

```
<owl:Class rdf:ID="PointBelongingToMagnitude">
    <owl:equivalentClass>
        <owl:Class>
            <owl:intersectionOf rdf:parseType="Collection">
                <owl:Class rdf:about="#Point" />
                <owl:Restriction>
                    <owl:onProperty rdf:resource="#belongsToQuantitySpace" />
                    <owl:someValuesFrom>
                        <owl:Class>
                            <owl:intersectionOf rdf:parseType="Collection">
                                <owl:Class rdf:about="#QuantitySpace" />
                                <owl:Restriction>
                                    <owl:onProperty rdf:resource="#isQuantitySpaceOf" />
                                    <owl:someValuesFrom rdf:resource="#Magnitude" />
                                </owl:Restriction>
                            </owl:intersectionOf>
                        </owl:Class>
                    </owl:someValuesFrom>
                </owl:Restriction>
            </owl:intersectionOf>
        </owl:Class>
    </owl:equivalentClass>
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasInequality" />
            <owl:allValuesFrom rdf:resource="#PointBelongingToMagnitude" />
        </owl:Restriction>
    </rdfs:subClassOf>
    <owl:disjointWith rdf:resource="#PointBelongingToDerivative" />
</owl:Class>
<owl:Class rdf:ID="PointBelongingToDerivative">
    <owl:equivalentClass>
```

```
        <owl:Class>
            <owl:intersectionOf rdf:parseType="Collection">
                <owl:Class rdf:about="#Point" />
                <owl:Restriction>
                    <owl:onProperty rdf:resource="#belongsToQuantitySpace" />
                    <owl:someValuesFrom>
                        <owl:Class>
                            <owl:intersectionOf rdf:parseType="Collection">
                                <owl:Class rdf:about="#QuantitySpace" />
                                <owl:Restriction>
                                    <owl:onProperty rdf:resource="#isQuantitySpaceOf" />
                                    <owl:someValuesFrom rdf:resource="#Derivative" />
                                </owl:Restriction>
                            </owl:intersectionOf>
                        </owl:Class>
                    </owl:someValuesFrom>
                </owl:Restriction>
            </owl:intersectionOf>
        </owl:Class>
    </owl:equivalentClass>
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasInequality" />
            <owl:allValuesFrom rdf:resource="#PointBelongingToMagnitude" />
        </owl:Restriction>
    </rdfs:subClassOf>
    <owl:disjointWith rdf:resource="#PointBelongingToMagnitude" />
</owl:Class>
```

B.10.2 Inequalities Belonging to Magnitudes or Derivatives

```
<owl:ObjectProperty rdf:ID="hasInequalityTarget">
    <rdfs:domain rdf:resource="#Inequality" />
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="hasInequality">
    <rdfs:range rdf:resource="#Inequality" />
</owl:ObjectProperty>
<!-- Magnitudes can have inequalities with other magnitudes or
points (qualitative values) within their own quantity space as
targets -->
<owl:Class rdf:about="#Magnitude">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasInequality" />
            <owl:allValuesFrom>
                <owl:Class>
                    <owl:intersectionOf rdf:parseType="Collection">
                        <owl:Class rdf:about="#Inequality" />
                        <owl:Restriction>
                            <owl:onProperty rdf:resource="#hasInequalityTarget" />
                            <owl:allValuesFrom>
                                <owl:Class>
                                    <owl:unionOf rdf:parseType="Collection">
                                        <owl:Class rdf:about="#Magnitude" />
                                        <!-- The magnitude should not be the same individual as
                                        the domain. -->
                                        <owl:Class rdf:about="#Point" />
                                        <!-- The point should be in the same quantity space,
                                        but this is not expressable in OWL -->
                                    </owl:unionOf>
                                </owl:Class>
                            </owl:allValuesFrom>
                        </owl:Restriction>
                    </owl:intersectionOf>
                </owl:Class>
            </owl:allValuesFrom>
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
<!-- Derivatives can have inequalities with other derivatives or
points (qualitative values) within their own quantity space as
targets -->
<owl:Class rdf:about="#Derivative">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasInequality" />
            <owl:allValuesFrom>
                <owl:Class>
                    <owl:intersectionOf rdf:parseType="Collection">
                        <owl:Class rdf:about="#Inequality" />
                        <owl:Restriction>
                            <owl:onProperty rdf:resource="#hasInequalityTarget" />
                            <owl:allValuesFrom>
                                <owl:Class>
                                    <owl:unionOf rdf:parseType="Collection">
                                        <owl:Class rdf:about="#Derivative" />
                                        <!-- The derivative should not be the same individual
                                        as the domain, but that is not expressable in OWL. -->
                                        <owl:Class rdf:about="#Point" />
                                        <!-- The point should be in the quantity space of the
                                        derivative, but this is not expressable in OWL -->
                                    </owl:unionOf>
                                </owl:Class>
                            </owl:allValuesFrom>
                        </owl:Restriction>
                    </owl:intersectionOf>
                </owl:Class>
            </owl:allValuesFrom>
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
```

B.10.3 Inequalities belonging to Intervals

```
<owl:Class rdf:about="#Interval">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasInequality" />
            <owl:allValuesFrom>
                <owl:Class>
                    <owl:intersectionOf rdf:parseType="Collection">
                        <owl:Class rdf:about="#Inequality" />
                        <owl:Restriction>
                            <owl:onProperty rdf:resource="#hasInequalityTarget" />
                            <owl:minCardinality rdf:datatype="&xsd;nonNegativeInteger">1
                            </owl:minCardinality>
                        </owl:Restriction>
                        <owl:Restriction>
                            <owl:onProperty rdf:resource="#hasInequalityTarget" />
                            <owl:allValuesFrom rdf:resource="#Point" />
                        </owl:Restriction>
                    </owl:intersectionOf>
                </owl:Class>
            </owl:allValuesFrom>
        </owl:Restriction>
    </rdfs:subClassOf>
```

```
</owl:Class>
```

B.10.4 hasValue Relations

```
<owl:ObjectProperty rdf:ID="hasValueTarget" />
<owl:Class rdf:about="#hasValue">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasValueTarget" />
            <owl:allValuesFrom rdf:resource="#QualitativeValue" />
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
```

B.11 Calculus Relations

```
<owl:ObjectProperty rdf:ID="hasLefthandSide">
    <owl:inverseOf rdf:resource="#isLefthandSideOf" />
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="isLefthandSideOf">
    <owl:inverseOf rdf:resource="#hasLefthandSide" />
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="hasRighthandSide" />
<owl:Class rdf:about="#Derivative">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#isLefthandSideOf" />
            <owl:allValuesFrom>
                <owl:Class>
                    <owl:intersectionOf rdf:parseType="Collection">
                        <owl:Class rdf:about="#Calculus" />
                        <owl:Restriction>
                            <owl:onProperty rdf:resource="#hasLefthandSide" />
                            <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">1
                            </owl:cardinality>
                        </owl:Restriction>
                        <owl:Restriction>
                            <owl:onProperty rdf:resource="#hasRighthandSide" />
                            <owl:allValuesFrom rdf:resource="#Derivative" />
                        </owl:Restriction>
                        <owl:Restriction>
                            <owl:onProperty rdf:resource="#hasRighthandSide" />
                            <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">1
                            </owl:cardinality>
                        </owl:Restriction>
                        <owl:Restriction>
                            <owl:onProperty rdf:resource="#hasInequality" />
                            <owl:allValuesFrom>
                                <owl:Class>
                                    <owl:intersectionOf rdf:parseType="Collection">
                                        <owl:Class rdf:about="#Inequality" />
                                        <owl:Restriction>
                                            <owl:onProperty rdf:resource="#hasInequalityTarget" />
                                            <owl:allValuesFrom rdf:resource="#Derivative" />
                                        </owl:Restriction>
                                    </owl:intersectionOf>
                                </owl:Class>
                            </owl:allValuesFrom>
                        </owl:Restriction>
                    </owl:intersectionOf>
                </owl:Class>
            </owl:allValuesFrom>
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="#Magnitude">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#isLefthandSideOf" />
            <owl:allValuesFrom>
                <owl:Class>
                    <owl:intersectionOf rdf:parseType="Collection">
                        <owl:Class rdf:about="#Calculus" />
                        <owl:Restriction>
                            <owl:onProperty rdf:resource="#hasLefthandSide" />
                            <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">1
                            </owl:cardinality>
                        </owl:Restriction>
                        <owl:Restriction>
                            <owl:onProperty rdf:resource="#hasRighthandSide" />
                            <owl:allValuesFrom>
                                <owl:Class>
                                    <owl:unionOf rdf:parseType="Collection">
                                        <owl:Class rdf:about="#Magnitude" />
                                        <owl:Class rdf:about="#PointBelongingToMagnitude" />
                                    </owl:unionOf>
                                </owl:Class>
                            </owl:allValuesFrom>
                        </owl:Restriction>
                        <owl:Restriction>
                            <owl:onProperty rdf:resource="#hasRighthandSide" />
                            <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">1
                            </owl:cardinality>
                        </owl:Restriction>
                        <owl:Restriction>
                            <owl:onProperty rdf:resource="#hasInequality" />
                            <owl:allValuesFrom>
                                <owl:Class>
                                    <owl:intersectionOf rdf:parseType="Collection">
                                        <owl:Class rdf:about="#Inequality" />
                                        <owl:Restriction>
                                            <owl:onProperty rdf:resource="#hasInequalityTarget" />
                                            <owl:allValuesFrom>
                                                <owl:Class>
                                                    <owl:unionOf rdf:parseType="Collection">
                                                        <owl:Class rdf:about="#Magnitude" />
                                                        <owl:Class rdf:about="#PointBelongingToMagnitude" />
                                                    </owl:unionOf>
                                                </owl:Class>
                                            </owl:allValuesFrom>
                                        </owl:Restriction>
                                    </owl:intersectionOf>
                                </owl:Class>
                            </owl:allValuesFrom>
                        </owl:Restriction>
                    </owl:intersectionOf>
                </owl:Class>
            </owl:allValuesFrom>
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="#Point">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#isLefthandSideOf" />
            <owl:allValuesFrom>
```

```
                <owl:Class>
                    <owl:intersectionOf rdf:parseType="Collection">
                        <owl:Class rdf:about="#Calculus" />
                        <owl:Restriction>
                            <owl:onProperty rdf:resource="#hasLefthandSide" />
                            <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">1
                            </owl:cardinality>
                        </owl:Restriction>
                        <owl:Restriction>
                            <owl:onProperty rdf:resource="#hasRighthandSide" />
                            <owl:allValuesFrom>
                                <owl:Class>
                                    <owl:unionOf rdf:parseType="Collection">
                                        <owl:Class rdf:about="#Magnitude" />
                                        <owl:Class rdf:about="#PointBelongingToMagnitude" />
                                    </owl:unionOf>
                                </owl:Class>
                            </owl:allValuesFrom>
                        </owl:Restriction>
                        <owl:Restriction>
                            <owl:onProperty rdf:resource="#hasRighthandSide" />
                            <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">1
                            </owl:cardinality>
                        </owl:Restriction>
                        <owl:Restriction>
                            <owl:onProperty rdf:resource="#hasInequality" />
                            <owl:allValuesFrom>
                                <owl:Class>
                                    <owl:intersectionOf rdf:parseType="Collection">
                                        <owl:Class rdf:about="#Inequality" />
                                        <owl:Restriction>
                                            <owl:onProperty rdf:resource="#hasInequalityTarget" />
                                            <owl:allValuesFrom>
                                                <owl:Class>
                                                    <owl:unionOf rdf:parseType="Collection">
                                                        <owl:Class rdf:about="#Magnitude" />
                                                        <owl:Class rdf:about="#PointBelongingToMagnitude" />
                                                    </owl:unionOf>
                                                </owl:Class>
                                            </owl:allValuesFrom>
                                        </owl:Restriction>
                                    </owl:intersectionOf>
                                </owl:Class>
                            </owl:allValuesFrom>
                        </owl:Restriction>
                    </owl:intersectionOf>
                </owl:Class>
            </owl:allValuesFrom>
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
```

B.12 Aggregate

B.12.1 Model Fragment

```
<owl:Class rdf:about="#ModelFragment">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasCondition" />
            <owl:allValuesFrom>
                <owl:Class>
                    <owl:unionOf rdf:parseType="Collection">
```

```
                        <owl:Class rdf:about="#Structural" />
                        <owl:Class rdf:about="#Behavioural" />
                        <owl:Class rdf:about="#AssumptionType" />
                        <owl:Class rdf:about="#Inequality" />
                        <owl:Class rdf:about="#Calculus" />
                        <owl:Class rdf:about="#ModelFragment" />
                    </owl:unionOf>
                </owl:Class>
            </owl:allValuesFrom>
        </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasConsequence" />
            <owl:allValuesFrom>
                <owl:Class>
                    <owl:unionOf rdf:parseType="Collection">
                        <owl:Class rdf:about="#Entity" />
                        <owl:Class rdf:about="#Configuration" />
                        <owl:Class rdf:about="#Attribute" />
                        <owl:Class rdf:about="#AttributeValue" />
                        <owl:Class rdf:about="#Behavioural" />
                        <owl:Class rdf:about="#Dependency" />
                    </owl:unionOf>
                </owl:Class>
            </owl:allValuesFrom>
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
<owl:ObjectProperty rdf:ID="hasCondition" />
<owl:ObjectProperty rdf:ID="hasConsequence" />
```

B.12.2 Static Fragment

```
<owl:Class rdf:about="#StaticFragment">
    <owl:equivalentClass>
        <owl:Class>
            <owl:intersectionOf rdf:parseType="Collection">
                <owl:Class rdf:about="#ModelFragment" />
                <owl:Class>
                    <owl:complementOf>
                        <owl:Restriction>
                            <owl:onProperty rdf:resource="#hasConsequence" />
                            <owl:someValuesFrom rdf:resource="#Influence" />
                        </owl:Restriction>
                    </owl:complementOf>
                </owl:Class>
                <owl:Class>
                    <owl:complementOf>
                        <owl:Restriction>
                            <owl:onProperty rdf:resource="#hasCondition" />
                            <owl:someValuesFrom rdf:resource="#Agent" />
                        </owl:Restriction>
                    </owl:complementOf>
                </owl:Class>
                <owl:Class>
                    <owl:complementOf>
                        <owl:Restriction>
                            <owl:onProperty rdf:resource="#hasConsequence" />
                            <owl:someValuesFrom rdf:resource="#Entity" />
                        </owl:Restriction>
                    </owl:complementOf>
                </owl:Class>
                <owl:Class>
                    <owl:complementOf>
                        <owl:Restriction>
                            <owl:onProperty rdf:resource="#hasConsequence" />
                            <owl:someValuesFrom rdf:resource="#Configuration" />
                        </owl:Restriction>
                    </owl:complementOf>
                </owl:Class>
            </owl:intersectionOf>
        </owl:Class>
    </owl:equivalentClass>
</owl:Class>
```

B.12.3 Process Fragment

```
<owl:Class rdf:about="#ProcessFragment">
    <owl:equivalentClass>
        <owl:Class>
            <owl:intersectionOf rdf:parseType="Collection">
                <owl:Class rdf:about="#ModelFragment" />
                <owl:Restriction>
                    <owl:onProperty rdf:resource="#hasConsequence" />
                    <owl:someValuesFrom rdf:resource="#Influence" />
                </owl:Restriction>
                <owl:Class>
                    <owl:complementOf>
                        <owl:Restriction>
                            <owl:onProperty rdf:resource="#hasCondition" />
                            <owl:someValuesFrom rdf:resource="#Agent" />
                        </owl:Restriction>
                    </owl:complementOf>
                </owl:Class>
            </owl:intersectionOf>
        </owl:Class>
    </owl:equivalentClass>
</owl:Class>
```

B.12.4 Agent Fragment

```
<owl:Class rdf:about="#AgentFragment">
    <owl:equivalentClass>
        <owl:Class>
            <owl:intersectionOf rdf:parseType="Collection">
                <owl:Class rdf:about="#ModelFragment" />
                <owl:Restriction>
                    <owl:onProperty rdf:resource="#hasCondition" />
                    <owl:someValuesFrom rdf:resource="#Agent" />
                </owl:Restriction>
            </owl:intersectionOf>
        </owl:Class>
    </owl:equivalentClass>
</owl:Class>
```

B.12.5 Scenario

```
<owl:Class rdf:about="#Scenario">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#hasFact" />
            <owl:allValuesFrom>
                <owl:Class>
                    <owl:unionOf rdf:parseType="Collection">
                        <owl:Class rdf:about="#Structural" />
                        <owl:Class rdf:about="#Behavioural" />
                        <owl:Class rdf:about="#AssumptionType" />
                        <owl:Class rdf:about="#Inequality" />
                        <owl:Class rdf:about="#Calculus" />
                    </owl:unionOf>
                </owl:Class>
            </owl:allValuesFrom>
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>
<owl:ObjectProperty rdf:ID="hasFact" />
```

B.13 Assumption

```
<owl:Class rdf:ID="Assumption">
    <rdfs:label xml:lang="en">Assumption</rdfs:label>
    <rdfs:comment xml:lang="en">
        An assumption is an assertion that something is true
    </rdfs:comment>
    <rdfs:subClassOf>
        <owl:Restriction>
                <owl:onProperty rdf:resource="&qr;isAssumptionRegarding" />
                <owl:maxCardinality rdf:datatype="&xsd;nonNegativeInteger"
            >1</owl:maxCardinality>
        </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="&qr;isAssumptionRegarding" />
            <owl:allValuesFrom>
                <owl:Class>
                    <owl:unionOf rdf:parseType="Collection">
                        <owl:Class rdf:about="&qr;Entity" />
                        <owl:Class rdf:about="&qr;Agent" />
                    </owl:unionOf>
                </owl:Class>
            </owl:allValuesFrom>
        </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf rdf:resource="#AssumptionType" />
</owl:Class>
<owl:ObjectProperty rdf:ID="isAssumptionRegarding" />
```

C Example Qualitative Model Definitions

C.1 Entity and Agent Definitions

```
<owl:Class rdf:about="&qrm;owl_ae_Object"
    rdfs:comment=""
    rdfs:label="Object">
  <rdfs:subClassOf rdf:resource="&qr;Entity"/>
  <owl:disjointWith rdf:resource="&qrm;owl_ae_Material"/>
</owl:Class>
<owl:Class rdf:about="&qrm;owl_ae_Container"
    rdfs:comment=""
    rdfs:label="Container">
  <rdfs:subClassOf rdf:resource="&qrm;owl_ae_Object"/>
  <owl:disjointWith rdf:resource="&qrm;owl_ae_Tube"/>
</owl:Class>
<owl:Class rdf:about="&qrm;owl_ae_Tube"
    rdfs:comment=""
    rdfs:label="Tube">
  <rdfs:subClassOf rdf:resource="&qrm;owl_ae_Object"/>
  <owl:disjointWith rdf:resource="&qrm;owl_ae_Container"/>
</owl:Class>
<owl:Class rdf:about="&qrm;owl_ae_Material"
    rdfs:comment=""
    rdfs:label="Material">
  <rdfs:subClassOf rdf:resource="&qr;Entity"/>
  <owl:disjointWith rdf:resource="&qrm;owl_ae_Object"/>
</owl:Class>
```

C.2 Configuration Definitions

```
<owl:Class rdf:about="&qrm;owl_c_Connected"
    rdfs:comment=""
    rdfs:label="Connected">
  <rdfs:subClassOf rdf:resource="&qr;Configuration"/>
  <owl:disjointWith rdf:resource="&qrm;owl_c_Contains"/>
</owl:Class>
<owl:Class rdf:about="&qrm;owl_c_Contains"
    rdfs:comment=""
    rdfs:label="Contains">
  <rdfs:subClassOf rdf:resource="&qr;Configuration"/>
  <owl:disjointWith rdf:resource="&qrm;owl_c_Connected"/>
</owl:Class>
```

C.3 Assumptions

```
<owl:Class rdf:about="&qrm;owl_ae_Containernotempty"
    rdfs:comment="Assume the container is not empty"
    rdfs:label="Containernotempty">
  <rdfs:subClassOf rdf:resource="&qr;Assumption"/>
</owl:Class>
```

C.4 Attribute Definition

```
<owl:Class rdf:about="&qrm;owl_a_Openorclosed"
    rdfs:comment=""
    rdfs:label="Openorclosed">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:allValuesFrom rdf:resource="&qrm;owl_av_OpenorclosedValue"/>
      <owl:onProperty rdf:resource="&qr;hasAttributeValue"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf rdf:resource="&qr;Attribute"/>
</owl:Class>
<owl:Class rdf:about="&qrm;owl_av_OpenorclosedValue"
    rdfs:label="OpenorclosedValue">
  <rdfs:subClassOf rdf:resource="&qr;AttributeValue"/>
  <owl:oneOf rdf:parseType="Collection">
    <rdf:Description rdf:about="&qrm;owl_av_Open"/>
    <rdf:Description rdf:about="&qrm;owl_av_Closed"/>
  </owl:oneOf>
</owl:Class>
<qrm:owl_av_OpenorclosedValue rdf:about="&qrm;owl_av_Open"
    rdfs:label="Open"/>
<qrm:owl_av_OpenorclosedValue rdf:about="&qrm;owl_av_Closed"
    rdfs:label="Closed"/>
<owl:AllDifferent>
  <owl:distinctMembers rdf:parseType="Collection">
    <rdf:Description rdf:about="&qrm;owl_av_Open"/>
    <rdf:Description rdf:about="&qrm;owl_av_Closed"/>
  </owl:distinctMembers>
</owl:AllDifferent>
```

C.5 Quantity Space Definitions

```
<owl:Class rdf:about="&qrm;owl_qs_Minzeroplus"
    rdfs:comment=""
    rdfs:label="Minzeroplus">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="&qr;containsQualitativeValue"/>
      <owl:someValuesFrom>
        <owl:Class>
          <owl:intersectionOf rdf:parseType="Collection">
            <rdf:Description rdf:about="&qrm;owl_qv_Min"/>
            <owl:Restriction>
              <owl:onProperty rdf:resource="&qr;hasInequality"/>
              <owl:someValuesFrom>
                <owl:Class>
                  <owl:intersectionOf rdf:parseType="Collection">
                    <rdf:Description rdf:about="&qr;SmallerThan"/>
                    <owl:Restriction>
                      <owl:onProperty rdf:resource="&qr;hasInequalityTarget"/>
                      <owl:someValuesFrom rdf:resource="&qrm;owl_qv_Zero"/>
                    </owl:Restriction>
                  </owl:intersectionOf>
                </owl:Class>
              </owl:someValuesFrom>
            </owl:Restriction>
          </owl:intersectionOf>
        </owl:Class>
      </owl:someValuesFrom>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="&qr;containsQualitativeValue"/>
      <owl:someValuesFrom rdf:resource="&qrm;owl_qv_Zero"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="&qr;containsQualitativeValue"/>
      <owl:someValuesFrom>
```

```
        <owl:Class>
          <owl:intersectionOf rdf:parseType="Collection">
            <rdf:Description rdf:about="&qrm;owl_qv_Plus"/>
            <owl:Restriction>
              <owl:onProperty rdf:resource="&qr;hasInequality"/>
              <owl:someValuesFrom>
                <owl:Class>
                  <owl:intersectionOf rdf:parseType="Collection">
                    <rdf:Description rdf:about="&qr;GreaterThan"/>
                    <owl:Restriction>
                      <owl:onProperty rdf:resource="&qr;hasInequalityTarget"/>
                      <owl:someValuesFrom rdf:resource="&qrm;owl_qv_Zero"/>
                    </owl:Restriction>
                  </owl:intersectionOf>
                </owl:Class>
              </owl:someValuesFrom>
            </owl:Restriction>
          </owl:intersectionOf>
        </owl:Class>
      </owl:someValuesFrom>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf rdf:resource="&qr;QuantitySpace"/>
  <owl:disjointWith rdf:resource="&qrm;owl_qs_Mzp"/>
  <owl:disjointWith rdf:resource="&qrm;owl_qs_Zeroplusmax"/>
</owl:Class>
<owl:Class rdf:about="&qrm;owl_qv_Plus"
    rdfs:label="Plus">
  <owl:disjointWith rdf:resource="&qrm;owl_qv_Max"/>
  <owl:disjointWith rdf:resource="&qrm;owl_qv_Min"/>
  <owl:disjointWith rdf:resource="&qrm;owl_qv_Zero"/>
  <rdfs:subClassOf rdf:resource="&qr;Interval"/>
</owl:Class>
<owl:Class rdf:about="&qrm;owl_qv_Min"
    rdfs:label="Min">
  <owl:disjointWith rdf:resource="&qrm;owl_qv_Max"/>
  <owl:disjointWith rdf:resource="&qrm;owl_qv_Plus"/>
  <owl:disjointWith rdf:resource="&qrm;owl_qv_Zero"/>
  <rdfs:subClassOf rdf:resource="&qr;Interval"/>
</owl:Class>
<owl:Class rdf:about="&qrm;owl_qv_Zero"
    rdfs:label="Zero">
  <owl:disjointWith rdf:resource="&qrm;owl_qv_Max"/>
  <owl:disjointWith rdf:resource="&qrm;owl_qv_Min"/>
  <owl:disjointWith rdf:resource="&qrm;owl_qv_Plus"/>
  <rdfs:subClassOf rdf:resource="&qr;Point"/>
</owl:Class>
```

C.6 Quantity Definitions

```
<owl:Class rdf:about="&qrm;owl_q_Flow"
    rdfs:comment=""
    rdfs:label="Flow">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="&qr;hasMagnitude"/>
      <owl:someValuesFrom>
        <owl:Class>
          <owl:intersectionOf rdf:parseType="Collection">
            <rdf:Description rdf:about="&qr;Magnitude"/>
            <owl:Restriction>
              <owl:allValuesFrom>
                <owl:Class>
                  <owl:unionOf rdf:parseType="Collection">
                    <rdf:Description rdf:about="&qrm;owl_qs_Minzeroplus"/>
                  </owl:unionOf>
                </owl:Class>
              </owl:allValuesFrom>
              <owl:onProperty rdf:resource="&qr;hasQuantitySpace"/>
            </owl:Restriction>
          </owl:intersectionOf>
        </owl:Class>
      </owl:someValuesFrom>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf rdf:resource="&qr;Quantity"/>
</owl:Class>
```

C.7 Model Fragment Definitions

```
<owl:Class rdf:about="&qrm;owl_ag_Fluidcontainer"
    qr:isActive="true"
    rdfs:comment=""
    rdfs:label="Fluidcontainer">
  <qr:hasCondition rdf:resource="&qrm;Derivative3"/>
  <qr:hasCondition rdf:resource="&qrm;Magnitude3"/>
  <qr:hasCondition rdf:resource="&qrm;owl_ae_Container1"/>
  <qr:hasCondition rdf:resource="&qrm;owl_ae_Fluid1"/>
  <qr:hasCondition rdf:resource="&qrm;owl_c_Contains1"/>
  <qr:hasCondition rdf:resource="&qrm;owl_q_Height1"/>
  <qr:hasCondition rdf:resource="&qrm;owl_qs_Mzp3"/>
  <qr:hasCondition rdf:resource="&qrm;owl_qs_Zeroplusmax3"/>
  <qr:hasCondition rdf:resource="&qrm;owl_qv_Max3"/>
  <qr:hasCondition rdf:resource="&qrm;owl_qv_Min3"/>
  <qr:hasCondition rdf:resource="&qrm;owl_qv_Plus5"/>
  <qr:hasCondition rdf:resource="&qrm;owl_qv_Plus6"/>
  <qr:hasCondition rdf:resource="&qrm;owl_qv_Zero5"/>
  <qr:hasCondition rdf:resource="&qrm;owl_qv_Zero6"/>
  <qr:hasConsequence rdf:resource="&qrm;Derivative1"/>
  <qr:hasConsequence rdf:resource="&qrm;Derivative2"/>
  <qr:hasConsequence rdf:resource="&qrm;EqualTo1"/>
  <qr:hasConsequence rdf:resource="&qrm;Magnitude1"/>
  <qr:hasConsequence rdf:resource="&qrm;Magnitude2"/>
  <qr:hasConsequence rdf:resource="&qrm;PositiveProportionality1"/>
  <qr:hasConsequence rdf:resource="&qrm;PositiveProportionality2"/>
  <qr:hasConsequence rdf:resource="&qrm;QuantitySpaceCorrespondence1"/>
  <qr:hasConsequence rdf:resource="&qrm;QuantitySpaceCorrespondence2"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_q_Amount1"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_q_Pressure1"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qs_Mzp1"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qs_Mzp2"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qs_Zeroplusmax1"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qs_Zeroplusmax2"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qv_Max1"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qv_Max2"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qv_Min1"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qv_Min2"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qv_Plus1"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qv_Plus2"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qv_Plus3"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qv_Plus4"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qv_Zero1"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qv_Zero2"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qv_Zero3"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qv_Zero4"/>
  <qr:hasDisplayOriginX>2</qr:hasDisplayOriginX>
  <qr:hasDisplayOriginY>2</qr:hasDisplayOriginY>
  <qr:hasEditSizeHeight>517</qr:hasEditSizeHeight>
  <qr:hasEditSizeWidth>641</qr:hasEditSizeWidth>
  <rdfs:subClassOf rdf:resource="&qr;StaticFragment"/>
</owl:Class>
<qrm:owl_ae_Fluid rdf:about="&qrm;owl_ae_Fluid1"
    qr:is_hidden_on_screen="false"
    rdfs:comment=""
    rdfs:label="Fluid">
  <qr:hasQuantity rdf:resource="&qrm;owl_q_Amount1"/>
  <qr:hasQuantity rdf:resource="&qrm;owl_q_Height1"/>
  <qr:hasQuantity rdf:resource="&qrm;owl_q_Pressure1"/>
  <qr:has_xposition_on_screen>189</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>144</qr:has_yposition_on_screen>
</qrm:owl_ae_Fluid>
<qrm:owl_ae_Container rdf:about="&qrm;owl_ae_Container1"
    qr:is_hidden_on_screen="false"
    rdfs:comment=""
    rdfs:label="Container">
  <qr:hasConfiguration rdf:resource="&qrm;owl_c_Contains1"/>
  <qr:has_xposition_on_screen>189</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>66</qr:has_yposition_on_screen>
</qrm:owl_ae_Container>
<qrm:owl_q_Amount rdf:about="&qrm;owl_q_Amount1"
    qr:is_hidden_on_screen="false"
    rdfs:comment=""
    rdfs:label="Amount">
  <qr:hasCausalDependency rdf:resource="&qrm;PositiveProportionality2"/>
  <qr:hasDerivative rdf:resource="&qrm;Derivative1"/>
  <qr:hasMagnitude rdf:resource="&qrm;Magnitude1"/>
  <qr:has_xposition_on_screen>75</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>189</qr:has_yposition_on_screen>
</qrm:owl_q_Amount>
<qr:Magnitude rdf:about="&qrm;Magnitude1"
    rdfs:label="Magnitude">
  <qr:hasQuantitySpace rdf:resource="&qrm;owl_qs_Zeroplusmax1"/>
</qr:Magnitude>
<qr:Derivative rdf:about="&qrm;Derivative1"
    qr:is_hidden_on_screen="false"
    rdfs:label="Derivative">
  <qr:hasQuantitySpace rdf:resource="&qrm;owl_qs_Mzp1"/>
  <qr:has_xposition_on_screen>56</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>252</qr:has_yposition_on_screen>
</qr:Derivative>
<qrm:owl_qs_Zeroplusmax rdf:about="&qrm;owl_qs_Zeroplusmax1"
    qr:is_hidden_on_screen="false"
    rdfs:label="Zeroplusmax">
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Max1"/>
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Plus1"/>
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Zero1"/>
  <qr:has_xposition_on_screen>80</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>234</qr:has_yposition_on_screen>
</qrm:owl_qs_Zeroplusmax>
<qrm:owl_qs_Mzp rdf:about="&qrm;owl_qs_Mzp1"
    qr:is_hidden_on_screen="false"
    rdfs:label="Mzp">
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Min1"/>
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Plus2"/>
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Zero2"/>
  <qr:has_xposition_on_screen>56</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>230</qr:has_yposition_on_screen>
</qrm:owl_qs_Mzp>
<qrm:owl_qv_Max rdf:about="&qrm;owl_qv_Max1"
```

```
    rdfs:label="Max"/>
<qrm:owl_qv_Plus rdf:about="&qrm;owl_qv_Plus1"
    rdfs:label="Plus">
  <qr:hasInequality rdf:resource="&qrm;GreaterThan1"/>
  <qr:hasInequality rdf:resource="&qrm;SmallerThan1"/>
</qrm:owl_qv_Plus>
<qrm:owl_qv_Zero rdf:about="&qrm;owl_qv_Zero1"
    rdfs:label="Zero"/>
<qr:SmallerThan rdf:about="&qrm;SmallerThan1"
    rdfs:label="SmallerThan">
  <qr:hasInequalityTarget rdf:resource="&qrm;owl_qv_Max1"/>
</qr:SmallerThan>
<qr:GreaterThan rdf:about="&qrm;GreaterThan1"
    rdfs:label="GreaterThan">
  <qr:hasInequalityTarget rdf:resource="&qrm;owl_qv_Zero1"/>
</qr:GreaterThan>
<qrm:owl_qv_Plus rdf:about="&qrm;owl_qv_Plus2"
    rdfs:label="Plus">
  <qr:hasInequality rdf:resource="&qrm;GreaterThan2"/>
</qrm:owl_qv_Plus>
<qrm:owl_qv_Zero rdf:about="&qrm;owl_qv_Zero2"
    rdfs:label="Zero"/>
<qrm:owl_qv_Min rdf:about="&qrm;owl_qv_Min1"
    rdfs:label="Min">
  <qr:hasInequality rdf:resource="&qrm;SmallerThan2"/>
</qrm:owl_qv_Min>
<qr:GreaterThan rdf:about="&qrm;GreaterThan2"
    rdfs:label="GreaterThan">
  <qr:hasInequalityTarget rdf:resource="&qrm;owl_qv_Zero2"/>
</qr:GreaterThan>
<qr:SmallerThan rdf:about="&qrm;SmallerThan2"
    rdfs:label="SmallerThan">
  <qr:hasInequalityTarget rdf:resource="&qrm;owl_qv_Zero2"/>
</qr:SmallerThan>
<qrm:owl_q_Pressure rdf:about="&qrm;owl_q_Pressure1"
    qr:is_hidden_on_screen="false"
    rdfs:comment=""
    rdfs:label="Pressure">
  <qr:hasDerivative rdf:resource="&qrm;Derivative2"/>
  <qr:hasMagnitude rdf:resource="&qrm;Magnitude2"/>
  <qr:has_xposition_on_screen>300</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>199</qr:has_yposition_on_screen>
</qrm:owl_q_Pressure>
<qr:Magnitude rdf:about="&qrm;Magnitude2"
    rdfs:label="Magnitude">
  <qr:hasQuantitySpace rdf:resource="&qrm;owl_qs_Zeroplusmax2"/>
</qr:Magnitude>
<qr:Derivative rdf:about="&qrm;Derivative2"
    qr:is_hidden_on_screen="false"
    rdfs:label="Derivative">
  <qr:hasQuantitySpace rdf:resource="&qrm;owl_qs_Mzp2"/>
  <qr:has_xposition_on_screen>294</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>258</qr:has_yposition_on_screen>
</qr:Derivative>
<qrm:owl_qs_Zeroplusmax rdf:about="&qrm;owl_qs_Zeroplusmax2"
    qr:is_hidden_on_screen="false"
    rdfs:label="Zeroplusmax">
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Max2"/>
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Plus3"/>
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Zero3"/>
  <qr:has_xposition_on_screen>327</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>238</qr:has_yposition_on_screen>
</qrm:owl_qs_Zeroplusmax>
```

```
<qrm:owl_qs_Mzp rdf:about="&qrm;owl_qs_Mzp2"
    qr:is_hidden_on_screen="false"
    rdfs:label="Mzp">
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Min2"/>
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Plus4"/>
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Zero4"/>
  <qr:has_xposition_on_screen>294</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>236</qr:has_yposition_on_screen>
</qrm:owl_qs_Mzp>
<qrm:owl_qv_Max rdf:about="&qrm;owl_qv_Max2"
    rdfs:label="Max"/>
<qrm:owl_qv_Plus rdf:about="&qrm;owl_qv_Plus3"
    rdfs:label="Plus">
  <qr:hasInequality rdf:resource="&qrm;GreaterThan3"/>
  <qr:hasInequality rdf:resource="&qrm;SmallerThan3"/>
</qrm:owl_qv_Plus>
<qrm:owl_qv_Zero rdf:about="&qrm;owl_qv_Zero3"
    rdfs:label="Zero"/>
<qr:SmallerThan rdf:about="&qrm;SmallerThan3"
    rdfs:label="SmallerThan">
  <qr:hasInequalityTarget rdf:resource="&qrm;owl_qv_Max2"/>
</qr:SmallerThan>
<qr:GreaterThan rdf:about="&qrm;GreaterThan3"
    rdfs:label="GreaterThan">
  <qr:hasInequalityTarget rdf:resource="&qrm;owl_qv_Zero3"/>
</qr:GreaterThan>
<qrm:owl_qv_Plus rdf:about="&qrm;owl_qv_Plus4"
    rdfs:label="Plus">
  <qr:hasInequality rdf:resource="&qrm;GreaterThan4"/>
</qrm:owl_qv_Plus>
<qrm:owl_qv_Zero rdf:about="&qrm;owl_qv_Zero4"
    rdfs:label="Zero"/>
<qrm:owl_qv_Min rdf:about="&qrm;owl_qv_Min2"
    rdfs:label="Min">
  <qr:hasInequality rdf:resource="&qrm;SmallerThan4"/>
</qrm:owl_qv_Min>
<qr:GreaterThan rdf:about="&qrm;GreaterThan4"
    rdfs:label="GreaterThan">
  <qr:hasInequalityTarget rdf:resource="&qrm;owl_qv_Zero4"/>
</qr:GreaterThan>
<qr:SmallerThan rdf:about="&qrm;SmallerThan4"
    rdfs:label="SmallerThan">
  <qr:hasInequalityTarget rdf:resource="&qrm;owl_qv_Zero4"/>
</qr:SmallerThan>
<qrm:owl_q_Height rdf:about="&qrm;owl_q_Height1"
    qr:is_hidden_on_screen="false"
    rdfs:comment=""
    rdfs:label="Height">
  <qr:hasCausalDependency rdf:resource="&qrm;PositiveProportionality1"/>
  <qr:hasDerivative rdf:resource="&qrm;Derivative3"/>
  <qr:hasMagnitude rdf:resource="&qrm;Magnitude3"/>
  <qr:has_xposition_on_screen>187</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>194</qr:has_yposition_on_screen>
</qrm:owl_q_Height>
<qr:Magnitude rdf:about="&qrm;Magnitude3"
    rdfs:label="Magnitude">
  <qr:hasInequality rdf:resource="&qrm;EqualTo1"/>
  <qr:hasQuantitySpace rdf:resource="&qrm;owl_qs_Zeroplusmax3"/>
</qr:Magnitude>

<qr:Derivative rdf:about="&qrm;Derivative3"
    qr:is_hidden_on_screen="false"
    rdfs:label="Derivative">
  <qr:hasQuantitySpace rdf:resource="&qrm;owl_qs_Mzp3"/>
  <qr:has_xposition_on_screen>171</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>253</qr:has_yposition_on_screen>
</qr:Derivative>
<qrm:owl_qs_Zeroplusmax rdf:about="&qrm;owl_qs_Zeroplusmax3"
    qr:is_hidden_on_screen="false"
    rdfs:label="Zeroplusmax">
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Max3"/>
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Plus5"/>
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Zero5"/>
  <qr:hasCorrespondence rdf:resource="&qrm;QuantitySpaceCorrespondence1"/>
  <qr:hasCorrespondence rdf:resource="&qrm;QuantitySpaceCorrespondence2"/>
  <qr:has_xposition_on_screen>200</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>234</qr:has_yposition_on_screen>
</qrm:owl_qs_Zeroplusmax>
<qrm:owl_qs_Mzp rdf:about="&qrm;owl_qs_Mzp3"
    qr:is_hidden_on_screen="false"
    rdfs:label="Mzp">
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Min3"/>
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Plus6"/>
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Zero6"/>
  <qr:has_xposition_on_screen>171</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>231</qr:has_yposition_on_screen>
</qrm:owl_qs_Mzp>
<qrm:owl_qv_Max rdf:about="&qrm;owl_qv_Max3"
    rdfs:label="Max"/>
<qrm:owl_qv_Plus rdf:about="&qrm;owl_qv_Plus5"
    rdfs:label="Plus">
  <qr:hasInequality rdf:resource="&qrm;GreaterThan5"/>
  <qr:hasInequality rdf:resource="&qrm;SmallerThan5"/>
</qrm:owl_qv_Plus>
<qrm:owl_qv_Zero rdf:about="&qrm;owl_qv_Zero5"
    rdfs:label="Zero"/>
<qr:SmallerThan rdf:about="&qrm;SmallerThan5"
    rdfs:label="SmallerThan">
  <qr:hasInequalityTarget rdf:resource="&qrm;owl_qv_Max3"/>
</qr:SmallerThan>
<qr:GreaterThan rdf:about="&qrm;GreaterThan5"
    rdfs:label="GreaterThan">
  <qr:hasInequalityTarget rdf:resource="&qrm;owl_qv_Zero5"/>
</qr:GreaterThan>
<qrm:owl_qv_Plus rdf:about="&qrm;owl_qv_Plus6"
    rdfs:label="Plus">
  <qr:hasInequality rdf:resource="&qrm;GreaterThan6"/>
</qrm:owl_qv_Plus>
<qrm:owl_qv_Zero rdf:about="&qrm;owl_qv_Zero6"
    rdfs:label="Zero"/>
<qrm:owl_qv_Min rdf:about="&qrm;owl_qv_Min3"
    rdfs:label="Min">
  <qr:hasInequality rdf:resource="&qrm;SmallerThan6"/>
</qrm:owl_qv_Min>
<qr:GreaterThan rdf:about="&qrm;GreaterThan6"
    rdfs:label="GreaterThan">
  <qr:hasInequalityTarget rdf:resource="&qrm;owl_qv_Zero6"/>
</qr:GreaterThan>
<qr:SmallerThan rdf:about="&qrm;SmallerThan6"
    rdfs:label="SmallerThan">
  <qr:hasInequalityTarget rdf:resource="&qrm;owl_qv_Zero6"/>
</qr:SmallerThan>
<qrm:owl_c_Contains rdf:about="&qrm;owl_c_Contains1"
    qr:is_hidden_on_screen="false"
    rdfs:comment=""
    rdfs:label="Contains">
  <qr:hasConfigurationTarget rdf:resource="&qrm;owl_ae_Fluid1"/>
  <qr:has_xposition_on_screen>210</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>113</qr:has_yposition_on_screen>
</qrm:owl_c_Contains>
<qr:PositiveProportionality rdf:about="&qrm;PositiveProportionality1"
    qr:is_hidden_on_screen="false"
    rdfs:comment=""
    rdfs:label="PositiveProportionality">
  <qr:hasCausalDependencyTarget rdf:resource="&qrm;owl_q_Pressure1"/>
  <qr:has_xposition_on_screen>250</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>191</qr:has_yposition_on_screen>
</qr:PositiveProportionality>
<qr:PositiveProportionality rdf:about="&qrm;PositiveProportionality2"
    qr:is_hidden_on_screen="false"
    rdfs:comment=""
    rdfs:label="PositiveProportionality">
  <qr:hasCausalDependencyTarget rdf:resource="&qrm;owl_q_Height1"/>
  <qr:has_xposition_on_screen>137</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>192</qr:has_yposition_on_screen>
</qr:PositiveProportionality>
<qr:QuantitySpaceCorrespondence rdf:about="&qrm;QuantitySpaceCorrespondence1"
    qr:isDirected="false"
    qr:isInverted="false"
    qr:is_hidden_on_screen="false"
    rdfs:comment=""
    rdfs:label="QuantitySpaceCorrespondence">
  <qr:hasCorrespondenceTarget rdf:resource="&qrm;owl_qs_Zeroplusmax2"/>
  <qr:has_xposition_on_screen>280</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>222</qr:has_yposition_on_screen>
</qr:QuantitySpaceCorrespondence>
<qr:QuantitySpaceCorrespondence rdf:about="&qrm;QuantitySpaceCorrespondence2"
    qr:isDirected="false"
    qr:isInverted="false"
    qr:is_hidden_on_screen="false"
    rdfs:comment=""
    rdfs:label="QuantitySpaceCorrespondence">
  <qr:hasCorrespondenceTarget rdf:resource="&qrm;owl_qs_Zeroplusmax1"/>
  <qr:has_xposition_on_screen>161</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>215</qr:has_yposition_on_screen>
</qr:QuantitySpaceCorrespondence>
<qr:EqualTo rdf:about="&qrm;EqualTo1"
    qr:is_hidden_on_screen="false"
    rdfs:comment=""
    rdfs:label="EqualTo">
  <qr:hasInequalityTarget rdf:resource="&qrm;Magnitude2"/>
  <qr:has_xposition_on_screen>244</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>214</qr:has_yposition_on_screen>
</qr:EqualTo>
```

C.8 Scenario Definitions

```
<owl:Class rdf:about="&qrm;owl_ag_Utube"
    qr:isActive="true"
    rdfs:comment=""
    rdfs:label="Utube">
  <qr:hasConsequence rdf:resource="&qrm;Derivative4"/>
  <qr:hasConsequence rdf:resource="&qrm;Derivative5"/>
  <qr:hasConsequence rdf:resource="&qrm;GreaterThan11"/>
  <qr:hasConsequence rdf:resource="&qrm;Magnitude4"/>
  <qr:hasConsequence rdf:resource="&qrm;Magnitude5"/>
  <qr:hasConsequence rdf:resource="&qrm;hasValue1"/>
  <qr:hasConsequence rdf:resource="&qrm;hasValue2"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_ae_Container11"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_ae_Container21"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_ae_Tube1"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_ae_Water11"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_ae_Water21"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_c_Connected1"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_c_Connected2"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_c_Contains2"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_c_Contains3"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_q_Height2"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_q_Height3"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qs_Mzp4"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qs_Mzp5"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qs_Zeroplusmax4"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qs_Zeroplusmax5"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qv_Max4"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qv_Max5"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qv_Min4"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qv_Min5"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qv_Plus10"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qv_Plus7"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qv_Plus8"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qv_Plus9"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qv_Zero10"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qv_Zero7"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qv_Zero8"/>
  <qr:hasConsequence rdf:resource="&qrm;owl_qv_Zero9"/>
  <qr:hasDisplayOriginX>0</qr:hasDisplayOriginX>
  <qr:hasDisplayOriginY>0</qr:hasDisplayOriginY>
  <qr:hasEditSizeHeight>400</qr:hasEditSizeHeight>
  <qr:hasEditSizeWidth>500</qr:hasEditSizeWidth>
  <rdfs:subClassOf rdf:resource="&qr;Scenario"/>
</owl:Class>
<qrm:owl_ae_Container rdf:about="&qrm;owl_ae_Container21"
    qr:is_hidden_on_screen="false"
    rdfs:comment=""
    rdfs:label="Container2">
  <qr:hasConfiguration rdf:resource="&qrm;owl_c_Connected1"/>
  <qr:hasConfiguration rdf:resource="&qrm;owl_c_Contains3"/>
  <qr:has_xposition_on_screen>266</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>117</qr:has_yposition_on_screen>
</qrm:owl_ae_Container>
<qrm:owl_ae_Tube rdf:about="&qrm;owl_ae_Tube1"
    qr:is_hidden_on_screen="false"
    rdfs:comment=""
    rdfs:label="Tube">
  <qr:has_xposition_on_screen>182</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>23</qr:has_yposition_on_screen>
</qrm:owl_ae_Tube>
<qrm:owl_ae_Water rdf:about="&qrm;owl_ae_Water11"
    qr:is_hidden_on_screen="false"
    rdfs:comment=""
    rdfs:label="Water1">
  <qr:hasQuantity rdf:resource="&qrm;owl_q_Height2"/>
  <qr:has_xposition_on_screen>62</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>202</qr:has_yposition_on_screen>
</qrm:owl_ae_Water>
<qrm:owl_ae_Water rdf:about="&qrm;owl_ae_Water21"
    qr:is_hidden_on_screen="false"
    rdfs:comment=""
    rdfs:label="Water2">
  <qr:hasQuantity rdf:resource="&qrm;owl_q_Height3"/>
  <qr:has_xposition_on_screen>266</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>202</qr:has_yposition_on_screen>
```

```
</qrm:owl_ae_Water>
<qrm:owl_ae_Container rdf:about="&qrm;owl_ae_Container11"
    qr:is_hidden_on_screen="false"
    rdfs:comment=""
    rdfs:label="Container1">
  <qr:hasConfiguration rdf:resource="&qrm;owl_c_Connected2"/>
  <qr:hasConfiguration rdf:resource="&qrm;owl_c_Contains2"/>
  <qr:has_xposition_on_screen>62</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>117</qr:has_yposition_on_screen>
</qrm:owl_ae_Container>
<qrm:owl_q_Height rdf:about="&qrm;owl_q_Height2"
    qr:is_hidden_on_screen="false"
    rdfs:comment=""
    rdfs:label="Height">
  <qr:hasDerivative rdf:resource="&qrm;Derivative4"/>
  <qr:hasMagnitude rdf:resource="&qrm;Magnitude4"/>
  <qr:has_xposition_on_screen>82</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>240</qr:has_yposition_on_screen>
</qrm:owl_q_Height>
<qr:Magnitude rdf:about="&qrm;Magnitude4"
    rdfs:label="Magnitude">
  <qr:hasInequality rdf:resource="&qrm;GreaterThan11"/>
  <qr:hasQuantitySpace rdf:resource="&qrm;owl_qs_Zeroplusmax4"/>
</qr:Magnitude>
<qr:Derivative rdf:about="&qrm;Derivative4"
    qr:is_hidden_on_screen="false"
    rdfs:label="Derivative">
  <qr:hasQuantitySpace rdf:resource="&qrm;owl_qs_Mzp4"/>
  <qr:has_xposition_on_screen>62</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>293</qr:has_yposition_on_screen>
</qr:Derivative>
<qrm:owl_qs_Zeroplusmax rdf:about="&qrm;owl_qs_Zeroplusmax4"
    qr:is_hidden_on_screen="false"
    rdfs:label="Zeroplusmax">
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Max4"/>
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Plus7"/>
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Zero7"/>
  <qr:has_xposition_on_screen>89</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>273</qr:has_yposition_on_screen>
</qrm:owl_qs_Zeroplusmax>
<qrm:owl_qs_Mzp rdf:about="&qrm;owl_qs_Mzp4"
    qr:is_hidden_on_screen="false"
    rdfs:label="Mzp">
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Min4"/>
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Plus8"/>
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Zero8"/>
  <qr:has_xposition_on_screen>62</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>271</qr:has_yposition_on_screen>
</qrm:owl_qs_Mzp>
<qrm:owl_qv_Max rdf:about="&qrm;owl_qv_Max4"
    rdfs:label="Max"/>
<qrm:owl_qv_Plus rdf:about="&qrm;owl_qv_Plus7"
    rdfs:label="Plus">
  <qr:hasInequality rdf:resource="&qrm;GreaterThan7"/>
  <qr:hasInequality rdf:resource="&qrm;SmallerThan7"/>
</qrm:owl_qv_Plus>
<qrm:owl_qv_Zero rdf:about="&qrm;owl_qv_Zero7"
    rdfs:label="Zero"/>
<qr:SmallerThan rdf:about="&qrm;SmallerThan7"
    rdfs:label="SmallerThan">
  <qr:hasInequalityTarget rdf:resource="&qrm;owl_qv_Max4"/>
</qr:SmallerThan>
<qr:GreaterThan rdf:about="&qrm;GreaterThan7"
```

```
    rdfs:label="GreaterThan">
  <qr:hasInequalityTarget rdf:resource="&qrm;owl_qv_Zero7"/>
</qr:GreaterThan>
<qrm:owl_qv_Plus rdf:about="&qrm;owl_qv_Plus8"
    rdfs:label="Plus">
  <qr:hasInequality rdf:resource="&qrm;GreaterThan8"/>
</qrm:owl_qv_Plus>
<qrm:owl_qv_Zero rdf:about="&qrm;owl_qv_Zero8"
    rdfs:label="Zero"/>
<qrm:owl_qv_Min rdf:about="&qrm;owl_qv_Min4"
    rdfs:label="Min">
  <qr:hasInequality rdf:resource="&qrm;SmallerThan8"/>
</qrm:owl_qv_Min>
<qr:GreaterThan rdf:about="&qrm;GreaterThan8"
    rdfs:label="GreaterThan">
  <qr:hasInequalityTarget rdf:resource="&qrm;owl_qv_Zero8"/>
</qr:GreaterThan>
<qr:SmallerThan rdf:about="&qrm;SmallerThan8"
    rdfs:label="SmallerThan">
  <qr:hasInequalityTarget rdf:resource="&qrm;owl_qv_Zero8"/>
</qr:SmallerThan>
<qrm:owl_q_Height rdf:about="&qrm;owl_q_Height3"
    qr:is_hidden_on_screen="false"
    rdfs:comment=""
    rdfs:label="Height">
  <qr:hasDerivative rdf:resource="&qrm;Derivative5"/>
  <qr:hasMagnitude rdf:resource="&qrm;Magnitude5"/>
  <qr:has_xposition_on_screen>256</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>238</qr:has_yposition_on_screen>
</qrm:owl_q_Height>
<qr:Magnitude rdf:about="&qrm;Magnitude5"
    rdfs:label="Magnitude">
  <qr:hasQuantitySpace rdf:resource="&qrm;owl_qs_Zeroplusmax5"/>
</qr:Magnitude>
<qr:Derivative rdf:about="&qrm;Derivative5"
    qr:is_hidden_on_screen="false"
    rdfs:label="Derivative">
  <qr:hasQuantitySpace rdf:resource="&qrm;owl_qs_Mzp5"/>
  <qr:has_xposition_on_screen>240</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>291</qr:has_yposition_on_screen>
</qr:Derivative>
<qrm:owl_qs_Zeroplusmax rdf:about="&qrm;owl_qs_Zeroplusmax5"
    qr:is_hidden_on_screen="false"
    rdfs:label="Zeroplusmax">
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Max5"/>
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Plus9"/>
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Zero9"/>
  <qr:has_xposition_on_screen>265</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>272</qr:has_yposition_on_screen>
</qrm:owl_qs_Zeroplusmax>
<qrm:owl_qs_Mzp rdf:about="&qrm;owl_qs_Mzp5"
    qr:is_hidden_on_screen="false"
    rdfs:label="Mzp">
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Min5"/>
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Plus10"/>
  <qr:containsQualitativeValue rdf:resource="&qrm;owl_qv_Zero10"/>
  <qr:has_xposition_on_screen>240</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>269</qr:has_yposition_on_screen>
</qrm:owl_qs_Mzp>
<qrm:owl_qv_Max rdf:about="&qrm;owl_qv_Max5"
    rdfs:label="Max"/>
<qrm:owl_qv_Plus rdf:about="&qrm;owl_qv_Plus9"
    rdfs:label="Plus">
  <qr:hasInequality rdf:resource="&qrm;GreaterThan9"/>
  <qr:hasInequality rdf:resource="&qrm;SmallerThan9"/>
</qrm:owl_qv_Plus>
<qrm:owl_qv_Zero rdf:about="&qrm;owl_qv_Zero9"
    rdfs:label="Zero"/>
<qr:SmallerThan rdf:about="&qrm;SmallerThan9"
    rdfs:label="SmallerThan">
  <qr:hasInequalityTarget rdf:resource="&qrm;owl_qv_Max5"/>
</qr:SmallerThan>
<qr:GreaterThan rdf:about="&qrm;GreaterThan9"
    rdfs:label="GreaterThan">
  <qr:hasInequalityTarget rdf:resource="&qrm;owl_qv_Zero9"/>
</qr:GreaterThan>
<qrm:owl_qv_Plus rdf:about="&qrm;owl_qv_Plus10"
    rdfs:label="Plus">
  <qr:hasInequality rdf:resource="&qrm;GreaterThan10"/>
</qrm:owl_qv_Plus>
<qrm:owl_qv_Zero rdf:about="&qrm;owl_qv_Zero10"
    rdfs:label="Zero"/>
<qrm:owl_qv_Min rdf:about="&qrm;owl_qv_Min5"
    rdfs:label="Min">
  <qr:hasInequality rdf:resource="&qrm;SmallerThan10"/>
</qrm:owl_qv_Min>
<qr:GreaterThan rdf:about="&qrm;GreaterThan10"
    rdfs:label="GreaterThan">
  <qr:hasInequalityTarget rdf:resource="&qrm;owl_qv_Zero10"/>
</qr:GreaterThan>
<qr:SmallerThan rdf:about="&qrm;SmallerThan10"
    rdfs:label="SmallerThan">
  <qr:hasInequalityTarget rdf:resource="&qrm;owl_qv_Zero10"/>
</qr:SmallerThan>
<qrm:owl_c_Connected rdf:about="&qrm;owl_c_Connected1"
    qr:is_hidden_on_screen="false"
    rdfs:comment=""
    rdfs:label="Connected">
  <qr:hasConfigurationTarget rdf:resource="&qrm;owl_ae_Tube1"/>
  <qr:has_xposition_on_screen>216</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>72</qr:has_yposition_on_screen>
</qrm:owl_c_Connected>
<qrm:owl_c_Connected rdf:about="&qrm;owl_c_Connected2"
    qr:is_hidden_on_screen="false"
    rdfs:comment=""
    rdfs:label="Connected">
  <qr:hasConfigurationTarget rdf:resource="&qrm;owl_ae_Tube1"/>
  <qr:has_xposition_on_screen>107</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>79</qr:has_yposition_on_screen>
</qrm:owl_c_Connected>
<qrm:owl_c_Contains rdf:about="&qrm;owl_c_Contains2"
    qr:is_hidden_on_screen="false"
    rdfs:comment=""
    rdfs:label="Contains">
  <qr:hasConfigurationTarget rdf:resource="&qrm;owl_ae_Water11"/>
  <qr:has_xposition_on_screen>80</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>163</qr:has_yposition_on_screen>
</qrm:owl_c_Contains>
<qrm:owl_c_Contains rdf:about="&qrm;owl_c_Contains3"
    qr:is_hidden_on_screen="false"
    rdfs:comment=""
    rdfs:label="Contains">
  <qr:hasConfigurationTarget rdf:resource="&qrm;owl_ae_Water21"/>
  <qr:has_xposition_on_screen>283</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>162</qr:has_yposition_on_screen>
</qrm:owl_c_Contains>
```

```
<qr:hasValue rdf:about="&qrm;hasValue1"
    rdfs:label="hasValue">
  <qr:hasValueTarget rdf:resource="&qrm;owl_qv_Plus9"/>
</qr:hasValue>
<qr:hasValue rdf:about="&qrm;hasValue2"
    rdfs:label="hasValue">
  <qr:hasValueTarget rdf:resource="&qrm;owl_qv_Plus7"/>
</qr:hasValue>
<qr:GreaterThan rdf:about="&qrm;GreaterThan11"
    qr:is_hidden_on_screen="false"
    rdfs:comment=""
    rdfs:label="GreaterThan">
  <qr:hasInequalityTarget rdf:resource="&qrm;Magnitude5"/>
  <qr:has_xposition_on_screen>195</qr:has_xposition_on_screen>
  <qr:has_yposition_on_screen>243</qr:has_yposition_on_screen>
</qr:GreaterThan>
```