CAFO: The Common Approach Foundation Ontology

Mark Fox[1], Anshula Chowdhury[2], Jane Zhang[3], Bart Gajderowicz[1], Tawfiq Abdulai[2], Daniela Rosu[1]

[1]Centre for Social Services Engineering
University of Toronto
csse.utoronto.ca

[2]Sametrica
sametrica.com

[3]Centre for Social Innovation
socialinnovation.org

Draft of: 15 November 2019

Technical Report
Centre for Social Services Engineering
University of Toronto
csse.utoronto.ca

Ontology URL: http://ontology.eil.utoronto.ca/CAO/cafo.owl#

Namespace: http://ontology.eil.utoronto.ca/CAO/cafo#

Suggested Prefix: cafo

Table of Contents

1 Introduction

This report defines the Common Approach **Foundation** Ontology (CAFO). It defines the foundational concepts that are used to construct an Ontology for the Theory of Change, including Activities, Indicators, Outcomes, and Stakeholders.

Concepts are formally defined as Ontologies (Gruber, 1993; Grüninger & Fox, 1995). This will:

- enable the representation of precise definitions thereby reducing the ambiguity of interpretation;
- make the components of impact computer interpretable so that open source software and other technologies developed for Big Data can be applied to analyze and interpret the data collected and generated by social enterprises;
- foster interoperability, i.e., the ability to understand and merge the information available from datasets spread across the social services community;
- automate the detection of inconsistencies in data, as well as the causes of the observed variations

The ontology is defined in Description Logic and published using the Semantic Web ontology language OWL.

The Core Common Approach Foundation Ontology (CAFO) combines, adapts and extends existing ontologies with additional concepts and axioms needed to represent the knowledge in the social purpose domain as captured by the Common Approach Framework. The remainder of this report specifies the foundation concepts upon which various theories of change can be defined. These include time, address, phone number, units of measure, person, activity, finance, organization and indicator.

We define the following prefixes that are used in the remainder of this report.

Prefix	URI
act	http://ontology.eil.utoronto.ca/tove/activity#
cafo	http://ontology.eil.utoronto.ca/CAO/cafo#
cav	http://ontology.eil.utoronto.ca/CAO/cav#
foaf	http://xmlns.com/foaf/0.1/
gcif	http://ontology.eil.utoronto.ca/GCI/Finance/GCI-Finance#
ic	http://ontology.eil.utoronto.ca/tove/icontact#
org	http://ontology.eil.utoronto.ca/organization#
time	http://www.w3.org/2006/time#
rel	http://purl.org/vocab/relationship/
sch	http://schema.org/
i72	http://ontology.eil.utoronto.ca/ISO21972/iso21972#
wgs84	http://www.w3.org/2003/01/geo/wgs84_pos#

2 Date/Time Ontology Pattern

2.1 Introduction

Time is both absolute and relative. It is important to understand not just at what time something occurred, but whether something occurred before, after or during some other event. To answer these questions, a much richer representation of time that supports reasoning about time points, time intervals and the relationships among them is needed. In summary the time ontology needs to be able to support the answering of questions such as:

- At what time did some activity or measurement occur?
- What was the duration of the activity?
- Did the activity occur before, after or during some other activity?

Many time ontologies have been developed. This document reuses "Time Ontology in OWL W3C Recommendation 19 October 2017" accessed at https://www.w3.org/TR/owl-time/.

2.2 Core Classes and Properties

Fundamental to any conceptual model is the time at which things occur. For example, questions may arise regarding the temporal relationship among measurements. Not just at what time something was measured, but whether it was measured before, after or during some event. For example, over what period of time were increases in indigenous women employment observed, and was that after the training program was completed? To answer these questions, a notion of time that supports reasoning about time points, time intervals and the relationships amongst them is needed. The following summarizes a subset of classes and relationships in OWL-Time.

There are three top level classes:

- **TemporalEntity**: It specifies the two types of time: Instant and Interval.
- **DateTimeDescription**: A specification of a date and time using a year, month, day, hour, etc. set of properties.
- **DurationDescription**: is a class that specifies a duration as any combination of years, weeks, days, hours, minutes, and seconds. Equivalent to ISO 19108 ‘TM_PeriodDuration’.

A TemporalEntity has 3 sub-classes:

- **Instant**: It represents a point in time. Equivalent to ISO 19108 ‘TM_Instant’.
- **Interval**: It represents a period of time with a beginning and an end. Equivalent to ISO 19108 ‘TM_Period’. If a DurationDescription is provided, then the difference between the beginning and end of the Interval should be equal to the DurationDescription.
- **ProperInterval:**It is an Interval where the beginning time is less than the end time.

A **TemporalEntity** has a beginning Instant, an ending Instant and a duration, which are denoted by the following properties:

- **hasBeginning**: links a TemporalEntity (domain) to an Instant (range) where the latter denotes the beginning of the TemporalEntity. Equivalent to ISO 19108 'Beginning'.
- **hasEnd:** links a TemporalEntity (domain) to an Instant (range) where the latter denotes the end of the TemporalEntity. Equivalent to ISO 19108 'Ending'.
- **hasDurationDescription**: links a TemporalEntity (domain) to an Interval (range) where the latter denotes the duration of the DurationDescription.

Finally, there is a set of properties that relate ProperInterval's, including intervalOverlaps, intervalAfter, intervalContains, etc. [1]

2.3 Graphical Depiction

The following directed graph (Figure 1) depicts the core classes that comprise OWL-Time.

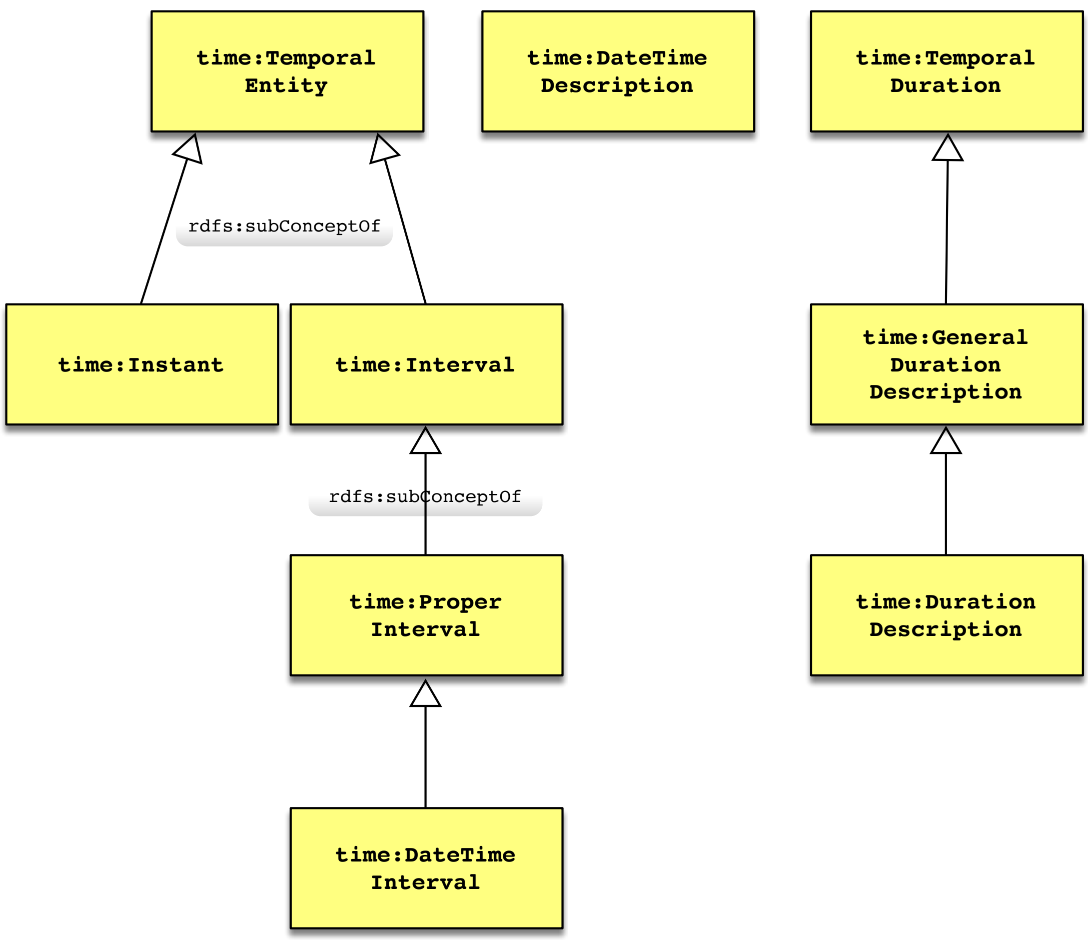

Figure 1: Time concepts

Figure 2 depicts the relationships that TemporalEntity has with the other classes.

[1] Since both OWL_time and ISO 19108 are based on Allen's temporal relations (Allen, 1983), each temporal relation in OWL-Time has an equivalent in ISO 19108.

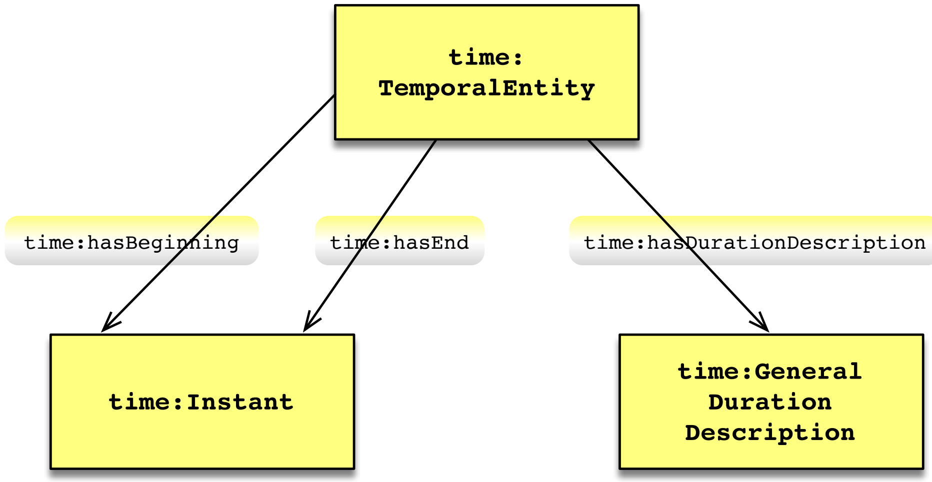

Figure 2: TemporalEntity relationships

2.4 Class Definitions

The following defines the basic classes and their properties. Additional properties in Appendix A.

Class	Property	Value Restriction
time:TemporalEntity	rdfs::subClassOf	time:TemporalThing
	time:hasBeginning	only 1 Instant
	time:hasEnd	only 1 Instant
	time:hasDurationDescription	only 1 DurationDescription
time:Instant	rdfs::subClassOf	time:TemporalEntity
	disjointWith	time:ProperInterval
time:Interval	rdfs::subClassOf	time:TemporalEntity
	time:inside	only time:Instant
time:ProperInterval	rdfs::subClassOf	time:Interval
	time:hasBeginning	exactly 1 time:before (self time:hasEnd)
	time:intervalBefore	only time:ProperInterval
	time:intervalAfter	only time:ProperInterval
	time:intervalDuring	only time:ProperInterval
	time:intervalContains	only time:ProperInterval
	time:intervalEqual	only time:ProperInterval
	time:intervalFinishes	only time:ProperInterval
	time:intervalFinishedBy	only time:ProperInterval
	time:intervalMeets	only time:ProperInterval
	time:intervalMetBy	only time:ProperInterval
	time:intervalOverlaps	only time:ProperInterval
	time:intervalOverlapedBy	only time:ProperInterval

	time:intervalStarts	only time:ProperInterval
	time:intervalStartedBy	only time:ProperInterval
time:DurationDescription	rdfs::subClassOf	time:TemporalThing
	time:years	max 1 xsd:nonNegativeInteger
	time:months	max 1 xsd:nonNegativeInteger
	time:weeks	max 1 xsd:nonNegativeInteger
	time:days	max 1 xsd:nonNegativeInteger
	time:hours	max 1 xsd:nonNegativeInteger
	time:minutes	max 1 xsd:nonNegativeInteger
	time:seconds	max 1 xsd:nonNegativeInteger
time:DateTimeDescription	rdfs::subClassOf	time:TemporalThing
	time:unitType	max 1 xsd:nonNegativeInteger
	time:timezone	max 1 TimeZone
	time:year	max 1 xsd:nonNegativeInteger
	time:month	max 1 xsd:nonNegativeInteger
	time:week	max 1 xsd:nonNegativeInteger
	time:day	max 1 xsd:nonNegativeInteger
	time:dayOfYear	max 1 xsd:nonNegativeInteger
	time:dayOfMonth	max 1 xsd:nonNegativeInteger
	time:dayOfWeek	max 1 xsd:nonNegativeInteger
	time:hour	max 1 xsd:nonNegativeInteger
	time:minute	max 1 xsd:nonNegativeInteger
	time:second	max 1 xsd:nonNegativeInteger
time:TemporalUnit	owl:equivalentClass	(time:unit time:Year, time:unitMonth, time:unitWeek, time:unitDay, time:unitHour, time:unitMinute, time:unitSecond)

3 Address Ontology Pattern

3.1 Introduction

The format of addresses across the globe vary in the types of information included. For example, a UK or Indian address may refer to a building name and a section of a city. The Address class defined here was designed to represent most of the types of addresses found by deconstructing it into its constituents.

3.2 Class Definition

The concept of Address class deconstructs the address into its constituents. Properties in blue are inherited from the superclass ic:Address.

Class	Property	Value Restriction
ic:Address	ic:hasAddressType	only ic:AddressType
	ic:hasStreetNumber	max 1 xsd:nonNegativeInteger
	ic:hasStreet	max 1 xsd:string
	ic:hasStreetType	max 1 ic:StreetType
	ic:hasStreetDirection	max 1 ic:StreetDirection
	ic:hasUnitNumber	max 1 xsd:nonNegativeInteger
	ic:hasPostalBox	max 1 xsd:string
	ic:hasBuilding	max 1 xsd:string
	ic:hasCitySection	max 1 xsd:string
	ic:hasCity	max 1 sc:City
	ic:hasProvince	max 1 sc:State
	ic:hasPostalCode	max 1 xsd:string
	ic:hasCountry (ISO 3166-2 alpha-2 2 letter country code)	max 1 schema:Country
	wgs84:lat	max 1 xsd:decimal
	wgs84:long	max 1 xsd:decimal
ic:AddressType	owl:equivalentTo	{cottage, home, work}
ic:StreetDirection	owl:equivalentTo	{east , north , south , west}
ic:StreetType	owl:equivalentTo	{avenue , boulevard , circle , crescent , drive , road , street}

4 Phone Number Ontology

The PhoneNumber class decomposes a phone number into its individual parts. Properties in blue are inherited from the superclass ic:PhoneNumber.

Class	Property	Value Restriction
ic:PhoneNumber	ic:hasCountryCode	exactly 1 xsd:nonNegativeInteger
	ic:hasAreaCode	exactly 1 xsd:nonNegativeInteger
	ic:hasPhoneNumber	exactly 1 xsd:nonNegativeInteger
	ic:hasPhoneType	exactly 1 PhoneType
ic:PhoneType	owl:equivalentTo	{cellPhone , faxPhone , homePhone , tollFreePhone , workPhone}

5 Measurement Ontology Patterns

5.1 Introduction

The purpose of a measurement ontology is to provide the underlying semantics of a number, such as what is being measured and the unit of measurement. A measurement ontology makes it possible to assure that numbers, such as indicators, are comparable, not that they are measuring the same thing, but the actual measures are of the same type, e.g., the counts of the homeless indigenous and non-indigenous populations, that comprise the ratio of homeless indigenous and non-indigenous population sizes, are of the same scale, for example, hundreds vs thousands.

5.2 Core Classes and Properties

The CAFO representation of measurement concepts reuses ISO 21972 which is based on the OM measurement ontology (Rijgersberg et al., 2013). The top row of Figure 3 depicts the basic classes of the measurement ontology. There are three main classes:

- a Quantity that denotes what is being measured, e.g., diameter of a ball, and links to the actual thing being measured via the phenomenon property, and the amount of the quantity via the value property that links to a Measure;
- a Unit_of_measure “is a definite magnitude of a quantity, defined and adopted by convention and/or by law. It is used as a standard for measurement of the same quantity, where any other value of the quantity can be expressed as a simple multiple of the unit of measure. For example, length is a quantity; the metre is a unit of length that represents a definite predetermined length. When we say 10 metre (or 10 m), we actually mean 10 times the definite predetermined length called ‘metre’." (OM, 2011).
- a Measure that denotes the value of the measurement (via the numerical_value property) which is linked to the both Quantity and Unit_of_measure.

For example, Indigenous Homeless Ratio is a subclass of Quantity that has a value that is a subclass of Measure whose units are a ‘population ratio unit’ that is an instance of Unit_of_measure. The actual value measured is a property of the Measure subclass ‘Indigenous homeless ratio measure’.

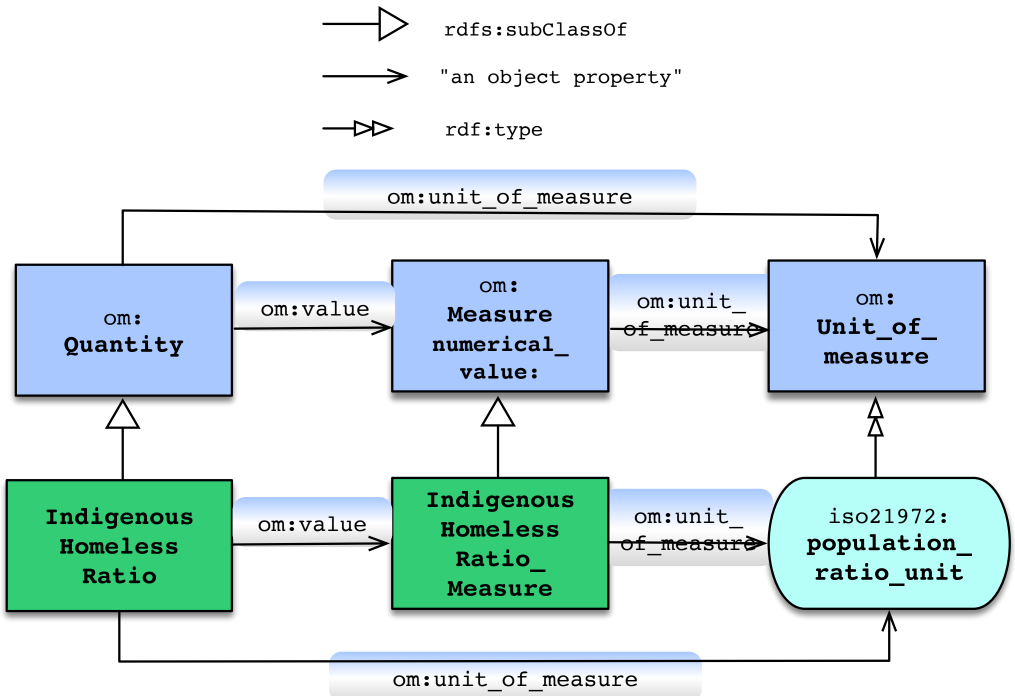

Figure 3: Basic Measurement Classes

The concept of a Quantity is common across many standards. BIPM (International Bureau of Weights and Measures) defines in JCGM 200:2012 that a quantity is a "property of a phenomenon, body, or substance, where the property has a magnitude that can be expressed as a number and a reference". W3C defines a Quantity in www.w3.org/2007/ont/unit as "A (scalar) physical quantity or dimensionless number". QUDT (http://qudt.org/schema/quantity) defines a Quantity to be "the measurement of an observable property of a particular object, event, or physical system." The definition of Quantity adopted in this document is the OM version defined above.

Unit_of_measure is divided into three subclasses:

- Singular_unit, such as a metre;
- Unit_multiple_or_submultiple, defines multiples or submultiples of a Singular_unit. For example, if the singular unit is a metre, then a kilometre would be a multiple, and a centimetre would be a submultiple, there are other possible multiples and submultiples. A Unit_multiple_or_submultiple links to the singular unit it is a multiple of via the singular_unit property;
- Compound_unit denotes a combination of Unit_of_measure's. For example, speed would be an instance of a Unit_division where singular (e.g., metre) or multiple unit (e.g., kilometer) would be divided by time (e.g., hour).

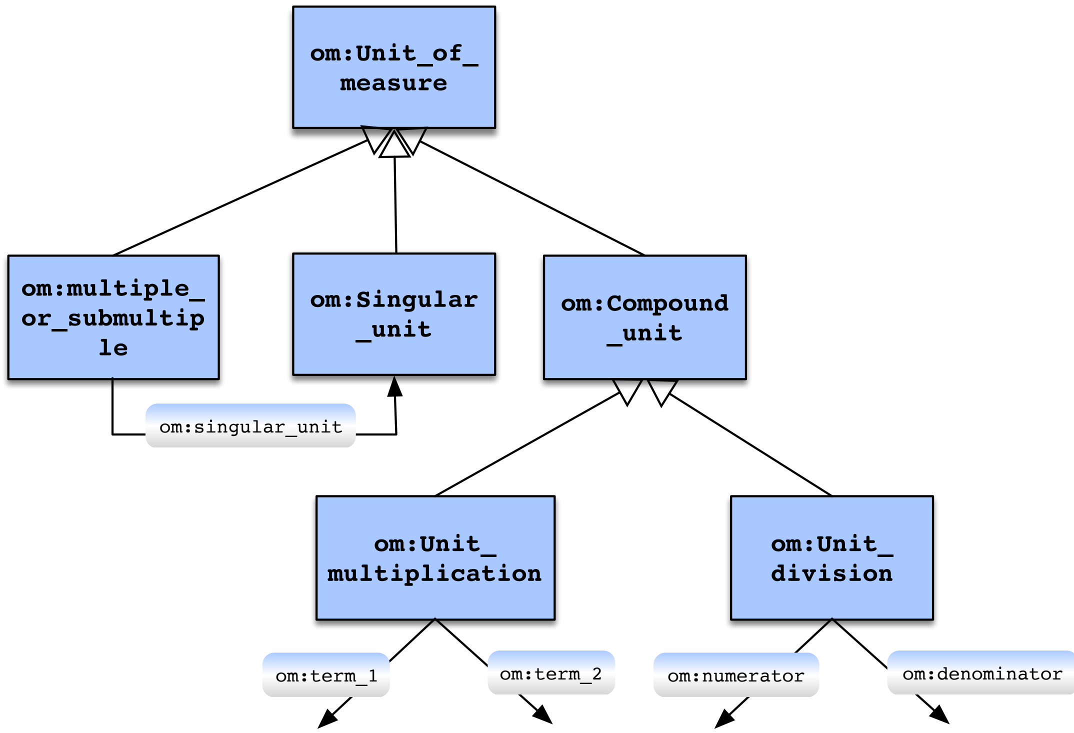

Figure 4: Unit of Measure Taxonomy

Defining a unit of measure not only requires the specification of whether it is singular or compound, but whether the scale of the unit is nominal, ordinal, interval or ratio. The latter two scales are also called cardinal scales. An example of a scale is the Celsius scale, a temperature scale. For ratio scales[2], a zero point can be defined.

[2] "Ratio data on the ratio scale has measurable intervals. For example, the difference between a height of six feet and five feet is the same as the interval between two feet and three feet. Where the ratio scale differs from the interval scale is that it also has a meaningful zero. The zero in a ratio scale means that something doesn't exist. For example, the zero in the Kelvin temperature scale means that heat does not exist at zero" (http://www.statisticshowto.com/ratio-scale/).

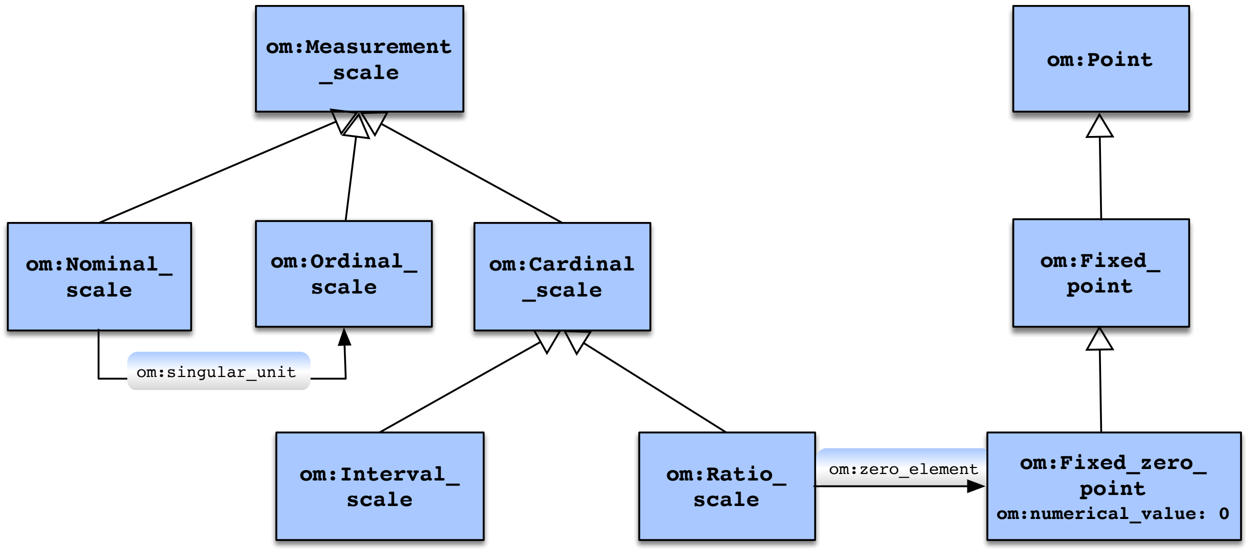

Figure 5: Measurement Scale Taxonomy

5.3 Formal Specification

The following tables formally define each of the classes of the measurement portion of this document.

Class	Property	Value Restriction
i72:Quantity	i72:value	exactly 1 i72:Measure
	i72:unit_of_measure	exactly 1 i72:Unit_of_measure
	i72:phenomenon	exactly 1 owl:Thing
i72:Measure	i72:unit_of_measure	exactly 1 i72:Unit_of_measure
	i72:numerical_value	exactly 1 xsd:string
i72:Unit_of_measure	rdfs::subClassOf	owl:Thing
i72:Singular_unit	rdfs:subClassOf	i72:Unit_of_measure
i72:Unit_multiple_or_sub multiple	rdfs:subClassOf	i72:Unit_of_measure
	i72:prefix	exactly 1 i72:Prefix
	i72:singular_unit	exactly 1 i72:Singular_unit
	i72:symbol	min 1 xsd:String
i72:Compound_unit	rdfs:subClassOf	i72:Unit_of_measure
i72:Unit_multiplication	rdfs:subClassOf	i72:Compound_unit
	i72:term_1	exatly 1 i72:Unit_of_measure
	i72:term_2	exactly 1 i72:Unit_of_measure
i72:Unit_division	rdfs:subClassOf	i72:Compound_unit
	i72:numerator	exactly 1 i72:Unit_of_measure
	i72:denominator	exactly 1 i72:Unit_of_measure
i72:Measurement_scale	rdfs:subClassOf	rdfs:Class
i72:Nominal_scale	rdfs:subClassOf	i72:Measurement_scale
i72:Ordinal_scale	rdfs:subClassOf	i72:Measurement_scale
i72:Cardinal_scale	rdfs:subClassOf	i72:Measurement_scale
i72:Interval_scale	rdfs:subClassOf	i72:Cardinal_scale
i72:Ratio_scale	rdfs:subClassOf	i72:Cardinal_scale
	i72:zero_element	exactly 1 i72:Fixed_zero_point
i72:Fixed_zero_point	rdfs:subClassOf	i72:Fixed_point
	numerical_value	i72:value "0"
i72:Fixed_point	rdfs:subClassOf	i72:Point
i72:Point	rdfs:comment	"A point is an element of an interval scale or a ratio scale, for example, 273.16 on the Kelvin scale indicates the triple point of water thermodynamic temperature. (OM, 2011)".

6 Indicator Measurement Ontology Pattern

6.1 Introduction

Indicators are used to measure Social Purpose Organizations' Outcomes. A challenge for Social Purpose Organizations is to define Indicators in a way that are precise, objective and auditable. Sadly, English is too imprecise a language for defining indicators such that they are consistently applied across Social Purpose Organizations. Consequently, the preferred approach is to formally represent the indicator definition using ontologies. If a definition changes, then the definition of the indicator is modified. If new indicators are introduced, then the definitions of the new indicators are constructed using the ontology. Using the ontology allows each Social Purpose Organization to define its own indicators while providing the analyst (human or machine) the ability to see the differences between the indicators[3]. In this section we present the core concepts for defining indicators.

6.2 Core Classes and Properties

The CAFO representation of indicator measurement concepts reuses ISO 21972 which is based on the Global City Indicator Foundation Ontology (Fox, 2013; 2105). An indicator is a quantity that is often a ratio of a numerator and denominator that are also quantities. It has a SE and time period associated with it. The numerator and denominator quantities can have different units of measure. One example of a unit of measure is the size of a population. A population_cardinality_unit is defined to be an individual of a Cardinality_unit that is a subclass of a Singular_unit. Figure 6 depicts the specification of the Cardinality_unit

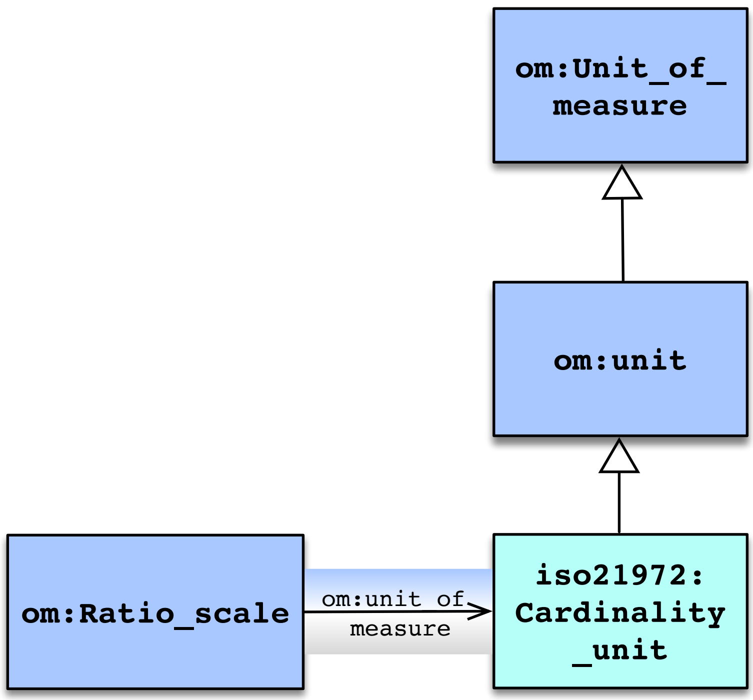

Figure 6: Cardinality_unit Definition

In Figure 7, population_cardinality_unit is depicted to be an instance of Cardinality_unit, which is the unit of measure for the cardinality of a set defined by a Population (defined in the next clause), and is associated with the symbol "pc". For example, 1100pc represents a population cardinality (or size) of 1100. This document takes advantage of prefix notations to scale the numbers by defining units of measures: kilopc, megapc and gigapc which are multiples of population_cardinality_unit. 1.1 kilopc represents 1100 pc.

[3] Note that there are two levels of definition: 1) as provided by the Indicator Repository Vocabulary which is integrated with the definition of Indicator in CACO, and 2) definition using the ontology in this section. The former supports the definition of the indicator using text (e.g., English), which can not be interpreted by software applications. The latter provides a computationallly precise definition that can be understood by both human and machine.

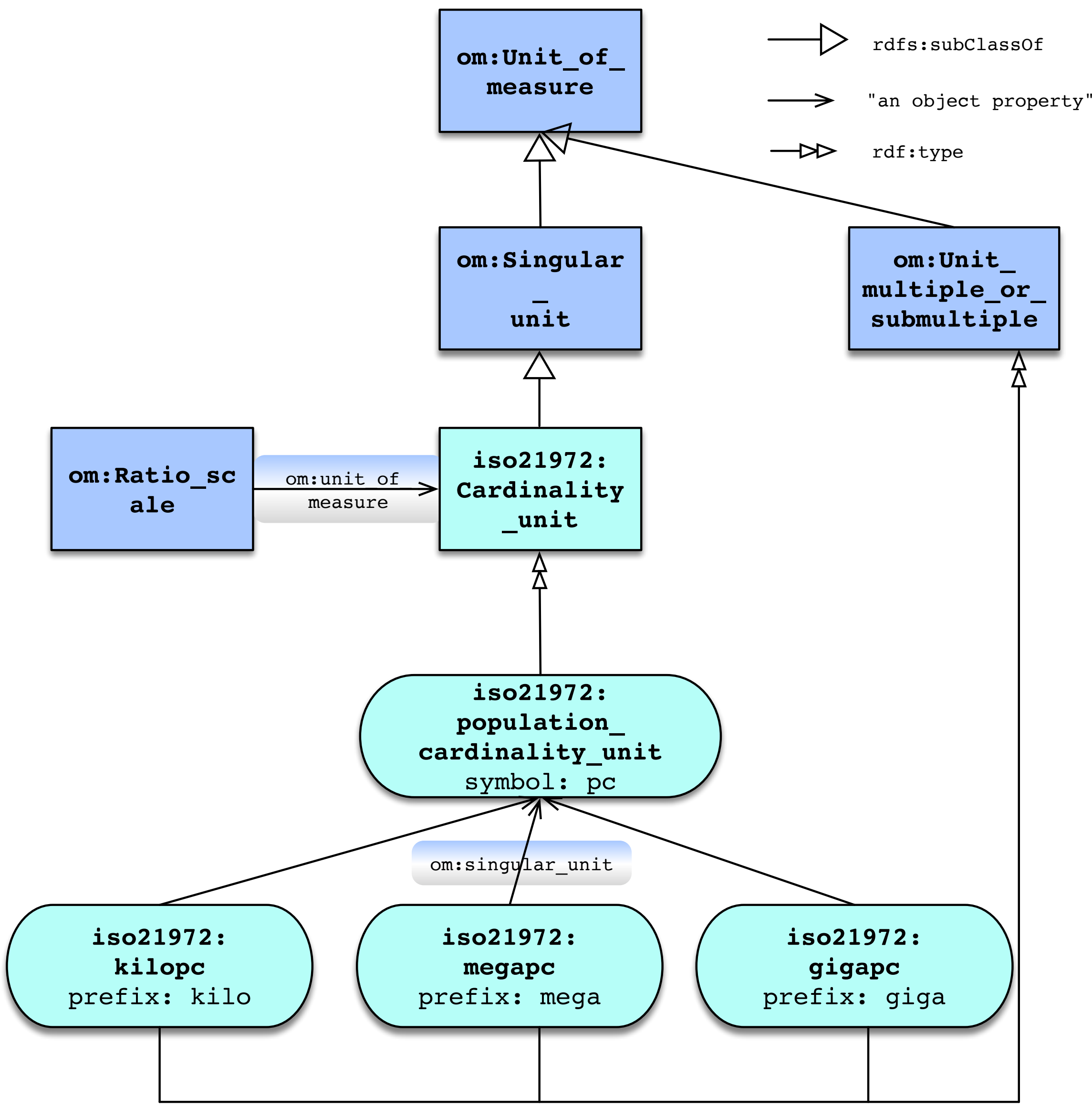

Figure 7: population_cardinality_unit Definition

With the above defined, it is possible to introduce the unit of measure for measuring a population ratio. population_ratio_unit is defined to be an instance of Unit_division. It has two properties:

- numerator whose range is restricted to being a population_cardinality_unit;
- denominator whose range is restricted to being a population_cardinality_unit.

In other words, a population ratio is the ratio of two population cardinalities (i.e., number of members/elements in each population).

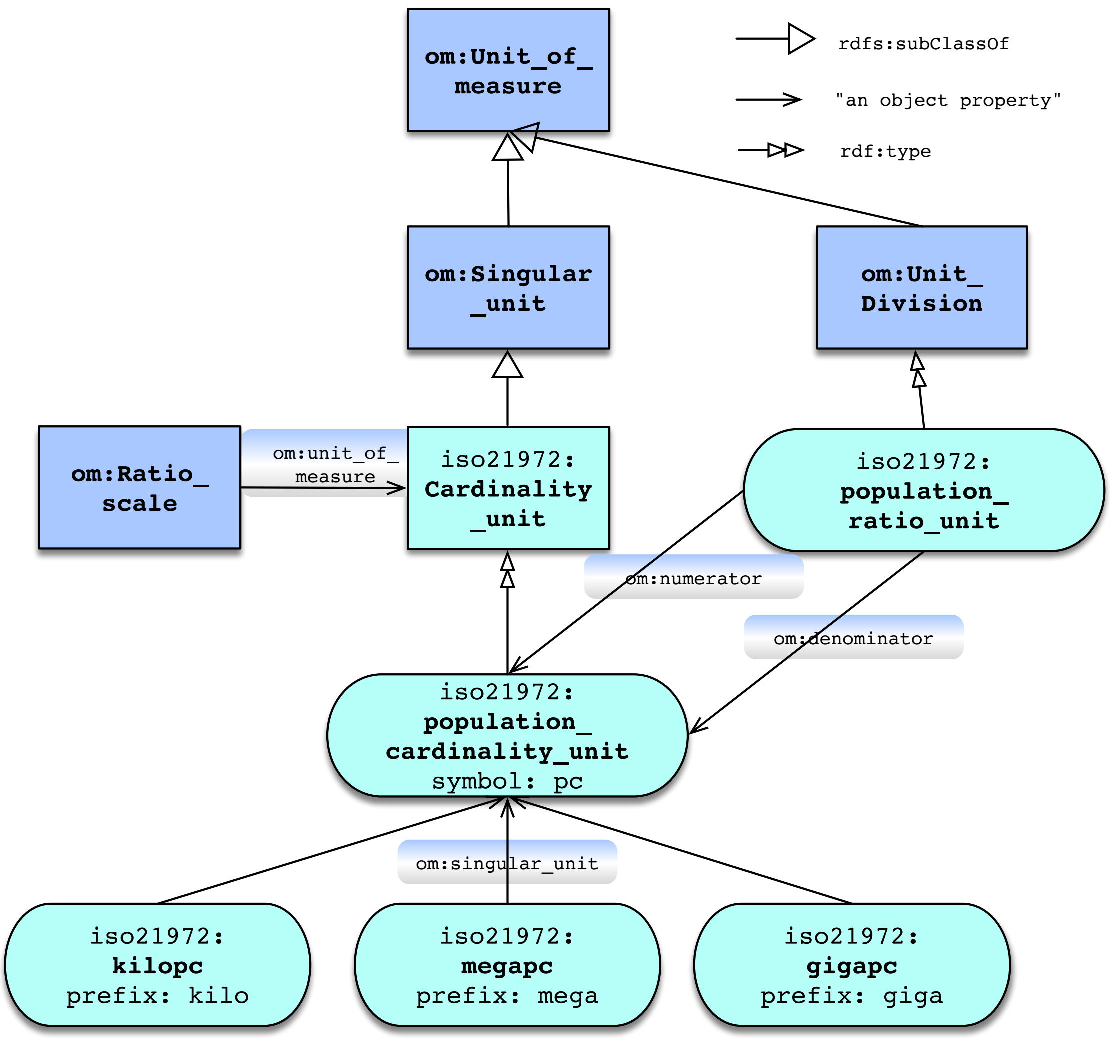

Figure 8: Depicts the population ratio unit definition

The above, provides the unit of measures for populations (pc) and population ratios (pc/pc).

6.3 Formal Specification

The following table specifies the key concepts for representing indicator definitions (using the Manchester syntax for Description Logic Full). Properties in blue are inherited from cav:Indicator. Properties in green are inherited from i72:Indicator.

Class	Property	Value
Indicator	rdfs:subClassOf	i72:Indicator
	rdfs:subClassOf	cav:Indicator

Class	Property	Value
	forOrganization	exactly 1 Organization
	i72:unit_of_measure	exactly 1 i72:Unit_of_measure
	i72:value	exactly 1 i72:Measure
	i72:for_time_interval	exactly 1 time:DateTImeInterval
	sch:identifier	exactly 1 xsd:string (begins with IND)
	sch:name	exactly 1 xsd:string
	sch:description	exactly 1 xsd:string
	definedBy	exactly 1 xsd:string (begins with ORG)
	forOutcomeS	only xsd:string (begins with OUT)
	hasSimilarIndicator	only xsd:string (begins with SI)
	sch:dateCreated	exactly 1 yyyy-mm-dd
RatioIndicator	rdfs:subClassOf	Indicator
	i72:unit_of_measure	i72:Unit_division
	i72:numerator	i72:Quantity
	i72:denominator	i72:Quantity

The basic definitions for population cardinality are as follows:

Class	Property	Value
i72:Cardinality_unit	rdfs::subClassOf	i72:Singular_unit
	inverse i72:unit_of_measure	exactly 1 i72:Cardinality_scale
i72:Cardinality_scale	rdfs::subClassOf	i72:Ratio_scale
	i72:zero_element	value fixed_zero_cardinality
i72:fixed_zero_cardinality	rdfs:type	i72:Fixed_zero_point
	i72:numerical_value	0
i72:population_cardinality_unit	rdfs:type	Cardinality_unit
	i72:symbol	"pc"

With the definition of a population_cardinality_unit, the different types of singular units of measures, and the compound units of measures upon which they are based on are defined (see Appendix 2). Note that the names of individuals of Monetary_unit adopt the ISO 4217 codes for currencies. Any new individuals of Monetary_unit should conform to the ISO 4217 standard. For Unit_multiple_or_submultiple individuals, we adopt ISO 80000 prefixes.

7 Organization Ontology Pattern

7.1 Introduction

Organizations provide another source of influence on the behaviour of Social Purpose Organizations. An organization is defined broadly as some group of individuals. The Organization ontology pattern captures the most common properties needed to represent SEs and other organizations they may work with. It also captures the structure of the organization and the roles within, such as CEO, Manager, Case Manager, etc. Roles further defined in the CACO-ORG Organization extensions (Fox et al., 2019b).

7.2 Core Classes and Properties

- Organization: A company or other sort of group of individuals in the urban system with some goal(s).
  An Organization may own Property, including different types of Buildings.
  An Organization may have an address.
  An Organization has at least 1 members.
  An Organization has some Goal(s); this represents some state or complex states, and allows for the representation of various groups' responsibilities.
  An Organization may be divided into Divisions.
- Organization Agent: Members of an organization.
  Organization Agents have goals, authority, and may be members of some team.
  An Organization Agent plays a Role within the Organization.
- Role: A Role has a single (possibly complex) Goal.
  A Role has some authority, requires some skill, and may also have some associated processes.

7.3 Formal Specification

The basic concept of an Organization is adapted from the TOVE Organization ontology (Fox et al., 1998). Following is a subset of the classes. Properties in blue are inherited from the superclass org:Organization. Note that we have included the string data properties from sch:Organization defined in the "Common Approach Indicator Vocabulary and Repository". Properties in green are inherited from org:Organization, and properties in blue from cav:Organization.

Class	Property	Value Restriction
Organization	rdfs:subClassOf	org:Organization
	rdfs:subClassOf	cav:Organization
	org:hasID	only org:OrganizationID
	ic:hasAddress	only ic:Address
	org:hasLegalName	exactly 1 xsd:string

	org:hasLegalStatus	only owl:Thing
	ic:hasTelephone	ionly c:PhoneNumber
	org:numberOfEmployees	exactly 1 xsd:nonNegativeNumber
	org:hasRole	only Role
	org:consistsOf	only org:Division
	sch:identifier	exactly 1 xsd:string (begins with ORG)
	sch:legalName	exactly 1 xsd:string
	sch:description	exactly 1 xsd:string
	sch:address	min 1 xsd:string
	sch:telephone	min 1 "nnn-nnn-nnn"
	cav:hasContact	min 1 xsd:string
	sch:email	min 1 xsd:string
	cav:hasSIndicatorReport	only xsd:string (each begins with IR)
	sch:dateCreated	exactly 1 "yyyy-mm-dd"
org:Role	org:hasAuthority	only org:Authority
	org:hasGoal	only org:Goal
	org:hasPolicy	only org:Policy
	org:hasResource	only org:Resource
	org:requiresSkill	only org:Skill
	org:specializedRoleOf	only org:Role
	org:subordinateOf	only org:Role
	org:hasProcess	only org:Process
org:OrganizationID	org:issuedBy	org:Organization
	sch:identifier	exactly 1 xsd:string
	sch:dateCreated	exactly 1 xsd:dateTime

8 Person Ontology Pattern

8.1 Introduction

The CAFO Person ontology pattern defines human stakeholders and the various relationships they form. The ontology can capture a great deal of detail about an individual, depending on the available data and needs of the application. Meeting data privacy requirements are assumed to be the responsibility of the data repository system.

8.2 Core Classes and Properties

A person can play a role at an organization. For any role, a person can be assigned an ID, with additional meta-data about the role they play. Properties are included for possible disabilities, disease, and immigration status, for use by the theories of change.

Class	Property	Range
Person	rdfs:subClassOf	sch:Person
	ic:hasAddress	only ic:Address
	ic:hasPhoneNumber	only ic:PhoneNumber
	ic:hasEmail	only xsd:string x.x.x.x@x.x
	sc:birthDate	max 1 xsd:dateTime
	foaf:givenName	max 1 xsd:string
	foaf:middleName	max 1 xsd:string
	foaf:familyName	max 1 xsd:string
	foaf: formerName	only xsd:string
	hasGender	only cwrc:Gender
	hasEthnicity	only cwrc:Ethnicity
	hasReligeon	only cwrc:Religeon
	hasOccupation	only Occupation
	org:plays	only Role
	rel:spouseOf	max 1 Person
	hasDisability	only Disability
	hasDisease	only Disease
	hasImmigrationStatus	only ImmigrationStatus
	hasMaritalStatus	some MaritalStatus
	hasNumberofChildren	some xsd:nonNegativeInteger
	sch:indentifier	only xsd:string

9 Activity Ontology

9.1 Introduction

A stakeholder, whether an individual person or an organization, is able to perform certain tasks to achieve their goals. (Stakeholder is elaborated further in CACO) The CAFO adopts the TOVE Activity ontology from which the following text is extracted (Fox et al., 1994).

9.2 Core Classes and Properties

Action is represented by the combination of an activity and its corresponding enabling and caused states (Figure 8). An activity is the basic transformational action primitive with which processes and operations can be represented. An enabling state defines what has to be true of the world in order for the activity to be performed. A caused state defines what will be true of the world once the activity has been completed.

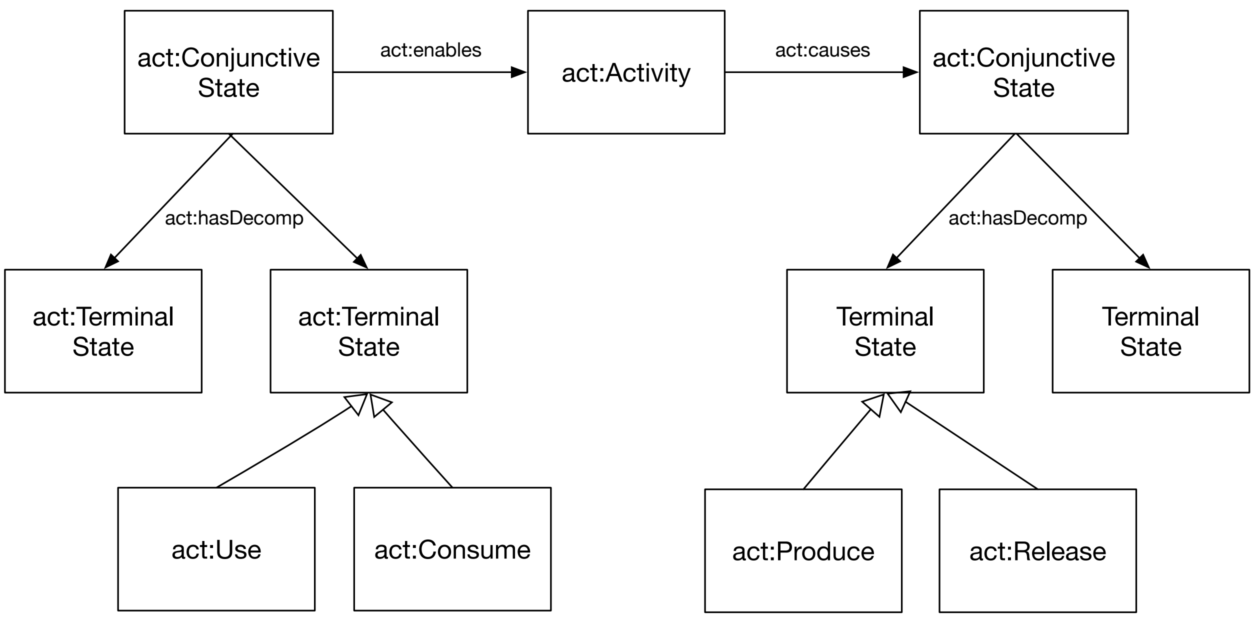

Figure 8: Activity-State Cluster

The status of an activity is reflected in an attribute called status. We define the domain of an activity's status as a set of linguistic constants.

- **dormant**: the activity is idle and has never been executing before.
- **enabled**: the activity is executing.
- **suspended**: the activity was executing and has been forced to an idle state.
- **reEnabled**: the activity is executing again.
- **completed**: the activity has finished.

An activity along with its enabling and caused states is called an activity cluster. The state tree linked by an enables relation to an activity specifies what has to be true in order for the activity to be performed. The state tree linked to an activity by a causes relation defines what will be true of the world once the activity has been completed.

There are two types of states: **terminal** and **non-terminal**.

Terminal States:

- **Use** signifies that a resource is to be used, but not consumed, by the activity, and will be released once the activity is completed.
- **Consume** signifies that a resource is to be used/consumed by the activity and will not exist once the activity is completed.
- **Release** signifies that a resource, which has been designated as being used is now available for use/consumption elsewhere.
- **Produce** signifies that a resource, that did not exist prior to the performance of the activity, has been created by the activity.
- **Predicate** represents an arbitrary predicate that evaluates to either true or false.

Non-terminal states: (allows for the boolean combination of states).

- **ConjunctiveState/DisjunctiveState** specifies that all substates / at least one substate must be satisfied.
- **Exclusive** specifies that only one substate must be satisfied.
- **Not** specifies that the substate must not be satisfied.
- **Composite Produce** signifies that a resource, that did not exist prior to the performance of the activity, has been created by the activity. This resource includes other materials that are only being used for a limited time and will be released by another activity. Except for the composite produce, each of these states can be further classified as being discrete or continuous.

Status: The status of a state is reflected in an attribute called status. We define the domain of a state's status as a set of linguistic constants. For example, the domain for discrete_consumption is:

- **possible/not_possible**: a unit of the resource that the state consumes is available/not available at the time required.
- **committed**: a unit of the resource that the state consumes has been reserved for consumption.
- **enabled**: a unit of the resource that the state consumes is being consumed.
- **completed**: unit of the resource that the state consumes had been consumed and is no longer needed.

We extend it by including properties for Outcome, Output and Input.

9.3 Formal Specification

Class	Property	Value Restriction
act:Activity	act:causes	max 1 act:State
	act:enabledBy	max 1 act:State
	act:hasElaboration	only act:Activity
	act:status	exactly 1 ActivityStatus
	act:initialActivity	max 1 act:Activity
	act:nextActivity	only act:Activity
	act:finalActivity	max 1 act:Activity
act:ActivityStatus	rdfs:subClassOf	act:Status
	owl:equivalentTo	{ act:completed, act:dormant, act:executing, act:reExecuting, act:suspended}
act:State	act:enables	only act:Activity
	act:causedBy	only act:Activity
	act:achievedAt	only time:TemporalEntity
act:TerminalState	rdfs:subClassOf	act:State
	owl:disjointWith	act:NonTerminalState
	act:hasDecomp	exactly 0 act:State
act:NonTerminalState	rdfs:subClassOf	act:State
	owl:disjointWtih	act:TerminalState
	act:hasDecomp	only act:State and min 1 act:State
act:ConjunctiveState	rdfs:subClassOf	act:NonTerminalState
	owl:disjointWith	act:DisjunctiveState
act:DisjunctiveState	rdfs:subClassOf	act:NonTerminalState
	owl:disjointWith	act:ConjunctiveState
act:Consume	rdfs:subClassOf	act:TerminalState
act:Produce	rdfs:subClassOf	act:TerminalState
act:Release	rdfs:subClassOf	act:TerminalState

10 Resource

The concept of a Resource is that it is anything that can be used, consumed or produced in the performance of an activity. A basic property of a resource is whether it can be divided and still be useable.

Class	Property	Value Restriction

Resource	hasType	{financial, skill, physical}
	isDivisible	only 1 xsd:boolean
FinancialResource	rdfs:subClassOf	Resource
SkillResource	rdfs:subClassOf	Resource
PhysicalResource	rdfs:subClassOf	Resource

11 Finance

The basis for the CAFO financial ontology is the GCI Finance Ontology (Wang & Fox, 2016). The following defines the Revenue classes:

Class	**Property**	**Range**
Revenue (Gap definition of revenue for non-profits)	rdfs:subClassOf	gcif:Revenue
	gcif:hasAmount	exactly 1 gcif:MonetaryValue
	gcif:hasCode	max 1 xsd:string
	gcif:hasSourcre	min 1 xsd:string
	hasDate	exactly 1 time:DateTime
TotalRevenue	rdfs:subClassOf	gci:GCI_Quantity
	forTimeInterval	only 1 time:DateTimeInterval
	gci:sum_of	only 1 TotalRevenuePopulation
TotalRevenuePopulation	rdfs:subClassOf	gci:GCI_population
	gs:defined_by	exactly 1 Revenue

Similarly Expenses is defined as:

Class	**Property**	**Range**
Expense	rdfs:subClassOf	gcif:Expenditure
	gcif:expensePurpose	min 1 xsd:string
	gcif:hasAmount	exactly 1 gcif:MonetaryValue
	gcif:hasCode	max 1 xsd:string
	gcif:hasSourcre	min 1 xsd:string
	forDate	exactly 1 xsd:DateTime
TotalExpense	rdfs:subClassOf	gci:GCI_quantity
	forTimeInterval	only 1 time:DateTimeInterval
	gci:sum_of	only 1 TotalExpensePopulation
TotalExpensePopulation	rdfs:subClassOf	gci:GCI_population
	gs:defined_by	exactly 1 Expense

12 Acknowledgements

This work was supported, in part, by Ontario Ministry of Economic Growth and Development and Employment and Social Development Canada.

13 References

Fox, M., Chionglo, J.F., and Fadel, F.G., (1993), "A Common Sense Model of the Enterprise", *Proceedings of the 2nd Industrial Engineering Research Conference*, pp. 425-429, Norcross GA: Institute for Industrial Engineers.

Fox, M.S., (2013), "A Foundation Ontology for Global City Indicators", Working Paper, Enterprise Integration Laboratory, University of Toronto, Revised: 13 October 2017

Fox, M.S., (2105), "The role of ontologies in publishing and analyzing city indicators", *Computers, Environment and Urban Systems*, Vol. 54, pp. 266–279. http://eil.mie.utoronto.ca/wp-content/uploads/2015/06/fox-ceus15.pdf.

Fox, M. S., Barbuceanu, M., & Gruninger, M. (1996). An organisation ontology for enterprise modeling: Preliminary concepts for linking structure and behaviour. *Computers in industry*, *29*(1-2), 123-134.

Fox, M.S., Chowdhury, A., Zhang, J., Gajderowicz, B., Abdulai, T., and Rosu, D., (2019a), "CAFO: The Common Approach Foundation Ontology for Modeling Theories of Change", Technical Report, Centre for Social Services Engineering, University of Toronto.

Fox, M.S., Chowdhury, A., Zhang, J., Gajderowicz, B., Abdulai, T., and Rosu, D., (2019b), "CACO-ORG: The Common Approach Core Ontology Organization Extension", Technical Report, Centre for Social Services Engineering, University of Toronto.

Fox, M.S., Chowdhury, A., Zhang, J., Gajderowicz, B., Abdulai, T., Ruff, K., and Rosu, D., (2019c), "CACO: The Common Approach Core Ontology for Modeling Theories of Change", Technical Report, Centre for Social Services Engineering, University of Toronto.

Fox, M.S., Gajderowicz, B., and Ruff, R., (2019d), "Common Approach Indicator Vocabulary and Repository", Technical Report, Centre for Social Services Engineering, University of Toronto.

Fox, M. S., & Gruninger, M. (1998). Enterprise modeling. *AI magazine*, *19*(3), 109-109.

Gajderowicz, B., Fox, M. S., & Grüninger, M. (2018). Ontology of social service needs: Perspective of a cognitive agent. In *Proceedings of the 2018 Joint Ontology WOrkshops, Cognition And OntologieS + Explainable AI* (pp. 1–12). Cape Town.

Gruber, T. R. (1993). Toward principles for the design of ontologies used for knowledge sharing. In *Proceedings of the international journal of human-computer studies*. Padova, Italy.

Grüninger, M., & Fox, M. S. (1995). Methodology for the Design and Evaluation of Ontologies. In *International Joint Conferences on Artificial Intelligence Workshop on Basic Ontological Issues in Knowledge Sharing*.

Ralser, T. (2008). Organizational Value/Nonprofit ROI. *ROI For Nonprofits: The New Key to Sustainability* (pp. 51–66). Hoboken, New Jersey: Wiley.

Rijgersberg, H., Van Assem, M., & Top, J. (2013). Ontology of units of measure and related concepts. *Semantic Web*, *4*(1), 3-13.

Wang, Z., and Fox, M.S., (2016), "A Finance Ontology for Global City Indicators (ISO 37120)", Working Paper, Enterprise Integration Laboratory, University of Toronto.

Appendix A. Date/Time Ontology

Property	Property	Value Restriction
time:hasDurationDescripition	rdf:type	owl:ObjectProperty
	rdfs:domain	time:TemporalEntity
	rdfs:range	time:DurationDescription
time:hasBeginning	rdf:type	owl:ObjectProperty
	rdfs:domain	time:TemporalEntity
	rdfs:range	time:Instant
time:hasEnd	rdf:type	owl:ObjectProperty
	rdfs:domain	time:TemporalEntity
	rdfs:range	time:Instant
time:before	rdf:type	owl:ObjectProperty
	rdfs:domain	time:TemporalEntity
	rdfs:range	time:TemporalEntity
	inverse	time:after
time:after	rdf:type	owl:ObjectProperty
	rdfs:domain	time:TemporalEntity
	rdfs:range	time:TemporalEntity
	inverse	time:before
time:intervalBefore	rdf:type	owl:ObjectProperty
	rdfs:domain	time:ProperInterval
	rdfs:range	time:ProperInterval
	inverse	time:intervalAfter

time:intervalDuring	rdf:type	owl:ObjectProperty
	rdfs:domain	time:ProperInterval
	rdfs:range	time:ProperInterval
	inverse	time:intervalContains
time:intervalContains	rdf:type	owl:ObjectProperty
	rdfs:domain	time:ProperInterval
	rdfs:range	time:ProperInterval
	inverse	time:intervalDuring
time:intervalEquals	rdf:type	owl:ObjectProperty
	rdfs:domain	time:ProperInterval
	rdfs:range	time:ProperInterval
time:intervalFinishes	rdf:type	owl:ObjectProperty
	rdfs:domain	time:ProperInterval
	rdfs:range	time:ProperInterval
	inverse	time:intervalFinishedBy
time:intervalFinishedBy	rdf:type	owl:ObjectProperty
	rdfs:domain	time:ProperInterval
	rdfs:range	time:ProperInterval
	inverse	time:intervalFinishes
time:intervalMeets	rdf:type	owl:ObjectProperty
	rdfs:domain	time:ProperInterval
	rdfs:range	time:ProperInterval
	inverse	time:intervalMetBy
time:intervalMetBy	rdf:type	owl:ObjectProperty
	rdfs:domain	time:ProperInterval
	rdfs:range	time:ProperInterval
	inverse	time:intervalMeets
time:intervalOverlaps	rdf:type	owl:ObjectProperty
	rdfs:domain	time:ProperInterval
	rdfs:range	time:ProperInterval
	inverse	time:intervalOverlapsBy
time:intervalOverlapedBy	rdf:type	owl:ObjectProperty
	rdfs:domain	time:ProperInterval
	rdfs:range	time:ProperInterval
	inverse	time:intervalOverlaps
time:intervalStarts	rdf:type	owl:ObjectProperty
	rdfs:domain	time:ProperInterval
	rdfs:range	ProperInterval
	inverse	time:intervalStartedBy
time:intervalStartedBy	rdf:type	owl:ObjectProperty
	rdfs:domain	time:ProperInterval
	rdfs:range	time:ProperInterval

	inverse	time:intervalStarts
time:inside	rdf:type	owl:ObjectProperty
	rdfs:domain	time:Interval
	rdfs:range	time:Instant
time:unitType	rdf:type	owl:ObjectProperty
	rdfs:domain	time:DateTimeDescription
	rdfs:range	time:TemporalUnit
time:timeZone	rdf:type	owl:ObjectProperty
	rdfs:domain	time:DateTimeDescription
	rdfs:range	tzont[4]:TimeZone

[4] tzont refers to the ontology defined in: http://www.w3.org/2006/timezone#

Appendix B. Unit of Measure Classes

With the definition of a population_cardinality_unit, the different types of singular units of measures, and the compound units of measures upon which they are based on are defined. Note that the names of individuals of Monetary_unit adopt the ISO 4217 codes for currencies.

Individual	Property	Value
i72:decibel	rdfs:type	i72:Singular_unit
i72:interruption	rdfs:type	i72:Singular_unit
i72:occurrence	rdfs:type	i72:Singular_unit
i72:gram	rdfs:type	i72:Singular_unit
i72:hour	rdfs:type	i72:Singular_unit
i72:watt	rdfs:type	i72:Singular_unit
i72:year	rdfs:type	i72:Singular_unit
i72:minute_time	rdfs:type	i72:Singular_unit
i72:metre_square	rdfs:type	i72:Unit_multiple
	i72:term_1	i72:metre
	i72:term_2	i72:metre
i72:kilopc (represents 1000 pc – population cardinality units)	rdfs:type	i72:Unit_multiple_or_submultiple
	i72:prefix	kilo
	i72:singular_unit	i72:population_cardinality_unit
	i72:symbol	"kilopc"
i72:megapc (represents 1,000,000 pc – population cardinality units)	rdfs:type	i72:Unit_multiple_or_submultiple
	i72:prefix	mega
	i72:singular_unit	i72:population_cardinality_unit
	i72:symbol	"megapc"
i72:gigapc (represents 10^9 pc – population cardinality units)	rdfs:type	i72:Unit_multiple_or_submultiple
	i72:prefix	giga
	i72:singular_unit	i72:population_cardinality_unit
	i72:symbol	"gigapc"
i72:hecto_kilo_pc (represents 100,000 pc – population cardinality units)	rdfs:type	i72:Unit_multiple
	i72:singular_unit	i72:population_cardinality_unit
	i72:prefix	gci:hecto_kilo
	i72:symbol	"hecto_kilo_pc"
i72:hecto_kilo (represents 100,000)	rdfs:type	i72:SI_prefix
i72:USD	rdfs:type	gci:Monetary_unit
i72:EUR	rdfs:type	gci:Monetary_unit

Any new individuals of Monetary_unit should conform to the ISO 4217 standard.