COMMON APPROACH
TO IMPACT MEASUREMENT

The Common Impact Data Standard:
An Ontology for Representing Impact

Mark Fox[1], Kate Ruff[2], Anshula Chowdhury[3], Bart Gajderowicz[1], Tawfiq Abdulai[3], Jane Zhang[4]

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

[2]Carleton Centre for Community Innovation
Carleton University

[3]Sametrica
sametrica.com

[4]Centre for Social Innovation
socialinnovation.org

Version 1.1, released: November 30, 2020

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

This document can be found at:
https://commonapproach.org/common-impact-data-standard/documentation
and
https://zenodo.org/record/4295989

Ontology URL: http://ontology.eil.utoronto.ca/cids/cids.owl

Namespace: http://ontology.eil.utoronto.ca/cids/cids#

Suggested Prefix: cids

For further information contact: Mark S. Fox, msf@eil.utoronto.ca

Recommended Attribution:
Fox et al. (2020) *The Common Impact Data Standard V1.1* The Common Approach to Impact Measurement.
DOI: 10.5281/zenodo.4295989

Revision	Date	Changes
1.1	November 20, 2020	Removed import of Common Approach Vocabulary (cav) and all uses of it in the ontology.
		Combined Foundation and Core Ontology into a single document now called the Common Impact Data Standard
		Added ImpactReport to separate reporting of indicators and outcomes.
		Moved “How Much” IMP information from StakeholderOutcome to the Outcome Report (Now ImpactReport) and added link from StakeholderOutcome to ImpactReport.
		hasSize in Outcome changed to hasImpactScale.
		hasSize, hasDepth and hasDuration properties moved to ImpactReport. For clarity, they were renamed ImpactSize ImpactDepth and ImpactDuration.
		Stakeholder properties added: sch:name (title), sch:description, hasCatchmentArea and hasStakeholderCharacteristic.
		hasThreshold and has Access added to Indicator.
		The value of i72:value property changed to i72:Measure
		hasAmount changed to i72:value in Output class.
	24 November 2020	Removed “benefitsFrom” property from ContributingStakeholder.
	26 November 2020	Added Quality measures to CIDS: Added ddqv:target only dqv:QualityPolicy to Indicator. Added dqv:hasQualityAnnotation, dqv:conformsTo and dqv:hasQualityMeasurement to IndicatorReport.
	28 November 2020	Replaced Dataset with dcat:Dataset. hasImportance in StakeHolderOutcome changed to a dataproperty. Deleted ImpactImportance class. Domain revised to support reference to an external standard/codelist – UNSDG example added. ImpactReport hasMethod replaced by prov:wasGeneratedBy. Change hasCatchmentArea to data property. Deleted CatchmentArea class. In Indicator, replaced hasDataSource with dqv:dataset; replaced Dataset class with dqv:Dataset; replaced hasMethod with prov:wasGeneratedBy:

Table of Contents

1. Introduction 4
1.1. Overview of the classes in the Common Impact Data Standard 4
1.2. Structure of this document 6
1.3. How to Read the Ontology Tables 6
2. Core Ontology: Common Impact Data Standard 9
2.1. ImpactModel 9
2.2. Organization Profile 11
2.3. Outcome, StakeholderOutcome, and ImpactReport 13
2.4. Stakeholder and Stakeholder Characteristics 16
2.5. Indicator and IndicatorReport 17
2.6. ImpactRisk 19
2.7. Program 20
2.8. Service 21
2.9. Activity 21
2.10. Input 22
2.11. Output 23
3. Foundation Ontology: Common Impact Data Standard 24
3.1. Date/Time – OWL-Time:2020 24
3.2. Address – tove/icontact 27
3.3. Phone Number – tove/icontact 28
3.4. Measurement – ISO/IEC 21972:2020 28
3.5. Indicator – ISO/IEC 21972:2020 32
3.6. Person 36
3.7. Activity – tove/activity 37
4. Example Use 41
4.1. cids:Outcome in JSON-LD 41
4.2. Cids:Outcome in Turtle 41
4.3. cids:Organization in JSON-LD 41
4.4. cids:Organization in Turtle 42
5. Acknowledgements 43
6. References 44

1. Introduction

The Common Impact Data Standard (CIDS) enables the exchange of impact information between organizations regardless of the impact models being used.

The Common Impact Data Standard is a way to represent impact as defined by the Impact Management Project. We understand Impact as a change in outcomes to people and the planet. To represent impact, the Common Impact Data Standard represents the five dimensions: what, who, how much, contribution and risk.

In addition, The Common Impact Data Standard represents Impact Models. We use impact models as a general term to describe a family of similar relational maps such as Logic model, Logical Framework Analysis, Theory of Change, Outcome Chain, Impact Map and Outcomes Map. CIDS also represents the processes by which a Social Purpose Organization delivers Outcomes to its Stakeholders. We call this the sixth dimension: How. This is illustrated in Figure 1.

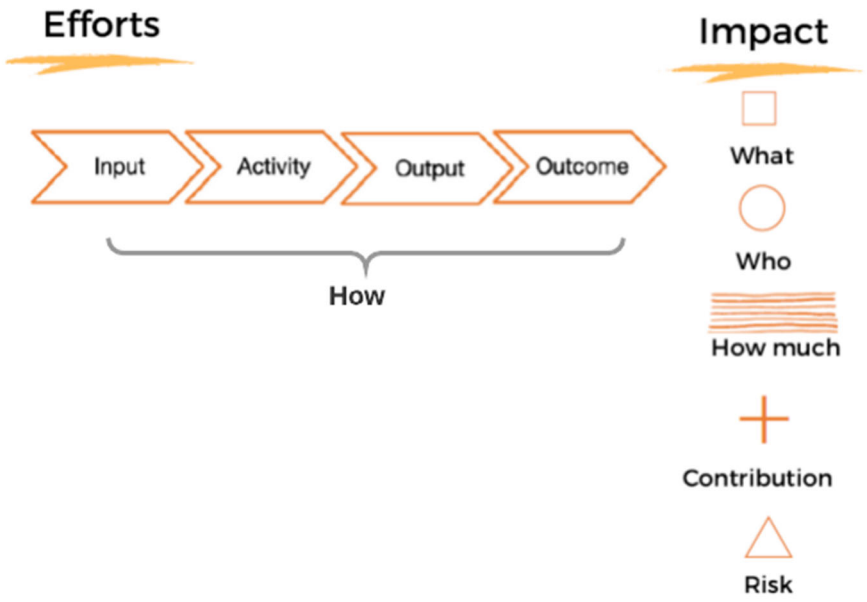

Figure 1: Summary of the impact representation specified in the Common Impact Data Standard. The Common Impact Data Standard represents the five dimensions of impact as defined by the Impact Management Project – what, who, how much, contribution and risk - plus a sixth dimension - how.

1.1. Overview of the classes in the Common Impact Data Standard

Technically speaking, the Common Impact Data Standard provides a core set of classes and properties that span all variations of impact models, making it possible for Social Purpose Organizations to share data amongst themselves regardless of the variation they use. While the core classes are adequate, for each of the impact models, further specialization of the core classes may be required to model the local variations. Figure 2 shows the classes and properties as they relate to the Impact Management Project five dimensions of impact.

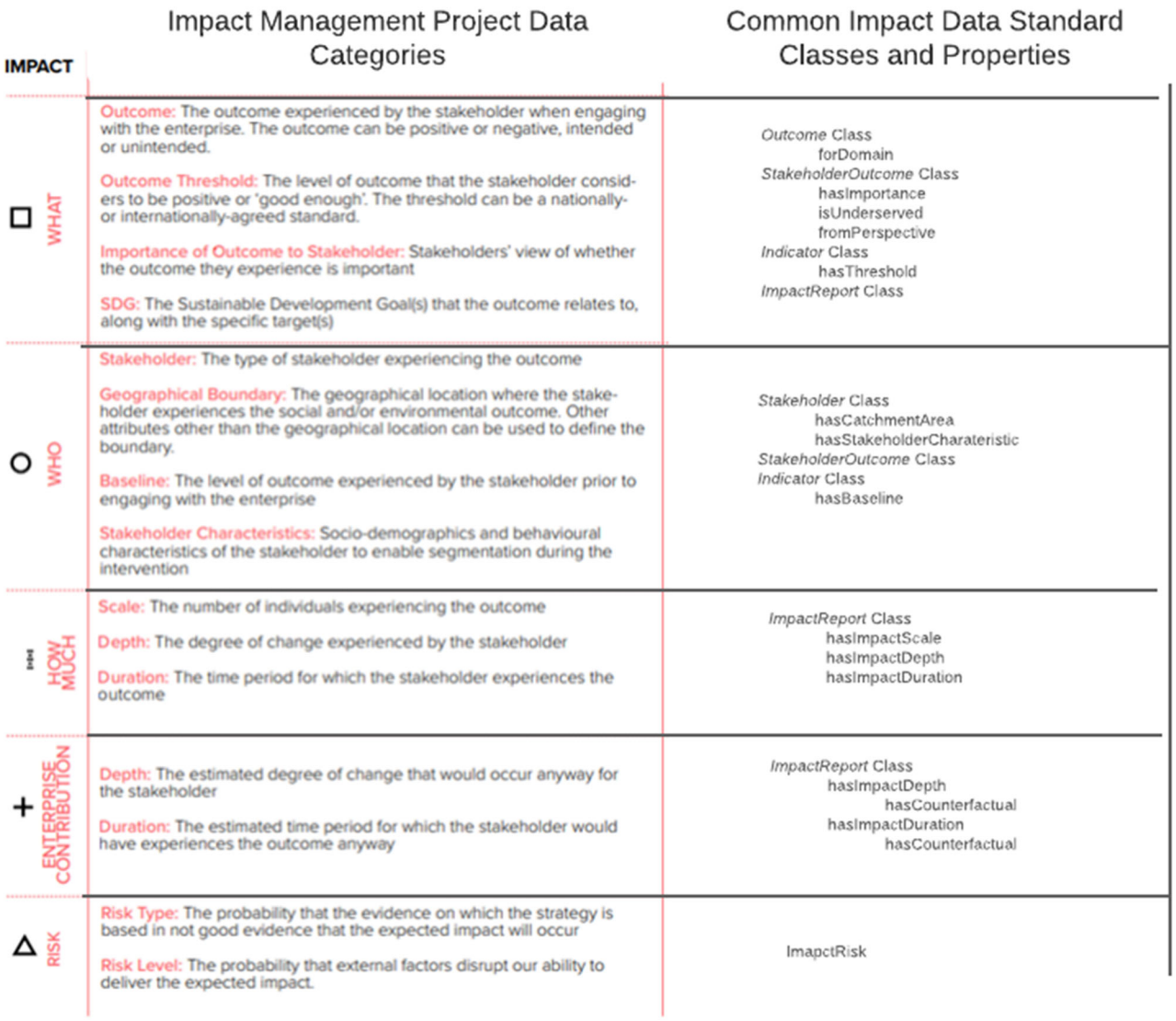

Figure 2: Impact Management Project and the Common Impact Data Standard

In order to represent the processes by which a Social Purpose Organization delivers Outcomes to its Stakeholders, we add a sixth dimension:

- **How**
  - *Program* Class
  - *Service* Class
  - *Activity* Class
  - *Input* Class
  - *Output* Class

1.2. Structure of this document

The Common Impact Data Standard is presented in two levels, as shown in Figure 3. The first, the Core Ontology, supports the representation of impact on people and the planet. It is applicable to both businesses and nonprofit organizations. The second level is referred to as the Foundation Ontology, which is a selection of existing standards and ontologies up with the Core Ontology is defined/grounded. Together, both ontologies form the Common Impact Data Standard.

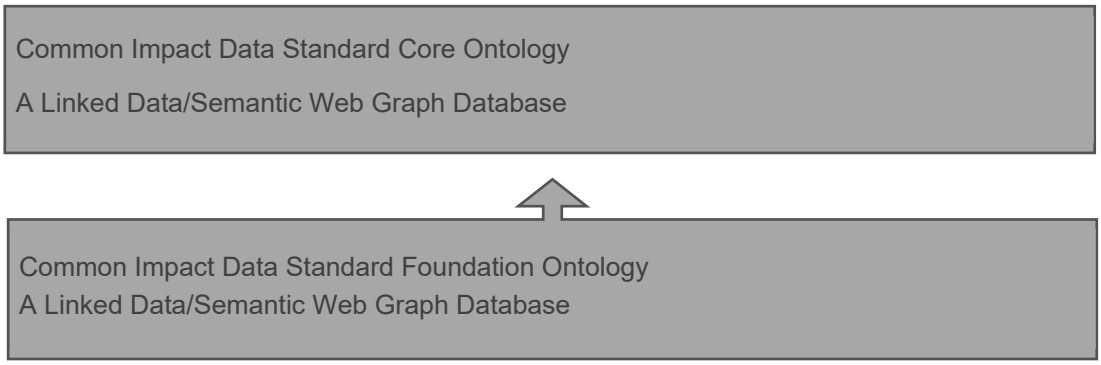

Figure 3: The Common Impact Data Standard is presented in two levels. There is a Core Ontology that is underpinned by a Foundation Ontology

This document defines core classes required to model various impact models. Specializations of these classes are defined in the following documents, which will be available at csse.utoronto.ca:

- CIDS-ACT: The Activity Extension.
- CIDS-FIN: The Finance Extension.
- CIDS -ORG: The Organization Extension.
- CIDS -OUT: The Outcome Extension.
- CIDS -PER: The Person Extension.

The classes included in this document do not represent a list of required classes that must be provided by a Social Purpose Organization, nor is it an extensive list of all classes that may exist. A separate document identifies which classes and properties are required at each level of adoption.

1.3. How to Read the Ontology Tables

The ontology specifies the “classes”, “properties” and possible restrictions on the “values” of a property within the context of a class. While it utilizes a simplified version of the Manchester syntax (Horridge et al., 2016) for Description Logic, formal definitions of classes are not provided, but can be found in the OWL files at http://ontology.eil.utoronto.ca/cids/cids.owl. The following table is an example. The class column specifies the name of the class being defined. This table describes the Organization class – it is a partial list for explanatory purposes. The class is defined to be a subclass of the **conjunction** of the properties in the property column. An organization is a subclass of owl:Thing. The property column specifies the properties used to define the class. Organization has properties including name, legal name, and address. The Value Restriction column specifies

the constraints on the values of each property for the class. The value of the “name” property must be a string. The value of the “address” property must be at most one instance of the class ic:Address.

Class	Property (partial list for example)	Value Restriction
Organization	rdfs:subClassOf	owl:Thing
	sch:name	only xsd:string
	sch:legalName	exactly 1 xsd:string
	ic:address	max 1 ic:Address

Table 1: Partial representation of the class Organization

The following value restrictions are used in this document:

- “min n”: Specifies that the property has to have a minimum n values.
- “max n”: Specifies that the property has to have a maximum n values.
- “exactly n”: Specifies that the property has to have exactly n values.
- “only”: Specifies that the values of the property can only be an instance/type of the class specified, e.g., a string, integer, or another class such as Organization.

We use camelCase for specifying classes, properties and instances. For example, “legalName” instead of “legal_name”. The first letter of a class name is capitalized. The first letter of a property and instance name are not capitalized.

An instance of a class must satisfy the class’s definition. The instance’s properties and values must satisfy the value restrictions of the class it is an instance of. Table 2 defines an instance of Organization. acmeSocialServices is of rdfs:type Organization. rdfs:type signifies that it is an instance. The remaining properties provide additional information about acmeSocialServices. Note that if we left out the property sch:legalName, it would be an error as the definition of Organization above has a value restriction for the property sch:legalName of having exactly 1 xsd:string. If zero or more than one was specified, it would be an error. The value of ic:address is NOT a string, but an instance of ic:Address class. In this example, properties have prefixes which identify the complete URI (i.e., namespace or library where the property name originates from).

Instance	Property	Value
acmeSocialServices	rdfs:type	caco:Organization
	sch:name	“Acme Social Services”
	sch:legalName	“Ontario 12345 Ltd.”
	ic:address	acmeAddress
acmeAddress	rdfs:type	ic:Address
	ic:hasStreet	“Bloor”
	ic:hasStreetType	ic:street
	ic:hasDirection	ic:west
	ic:hasStreetNumber	0

Table 2: An example of an Instance of an Organization

Table 3 lists the prefixes used in this document.

Prefix	URI
act	http://ontology.eil.utoronto.ca/tove/activity#
cids	http://ontology.eil.utoronto.ca/CIDS/cids#
dcat	http://www.w3.org/ns/dcat#
dqv	http://www.w3.org/ns/dqv#
ic	http://ontology.eil.utoronto.ca/tove/icontact#
i72	http://ontology.eil.utoronto.ca/ISO21972/iso21972#
oep	http://www.w3.org/2001/sw/BestPractices/OEP/SimplePartWhole/part.owl#
rdfs	http://www.w3.org/2000/01/rdf-schema#
sch	http://schema.org/
sur	http://ontology.eil.utoronto.ca/tove/survey#
time	https://www.w3.org/2006/time#
xsd	http://www.w3.org/2001/XMLSchema#

Table 3: Prefixes used in this document and the connected URI

2. Core Ontology: Common Impact Data Standard

In this section, we defined the classes that comprise the Common Impact Data Standard Core Ontology. Figure 4 depicts the main classes and a subset of the properties that connect them. The classes and properties together are sufficient for representing all impact models. Classes in white define the impact model, classes in green report on the performance of the organization with respect to their impact model.

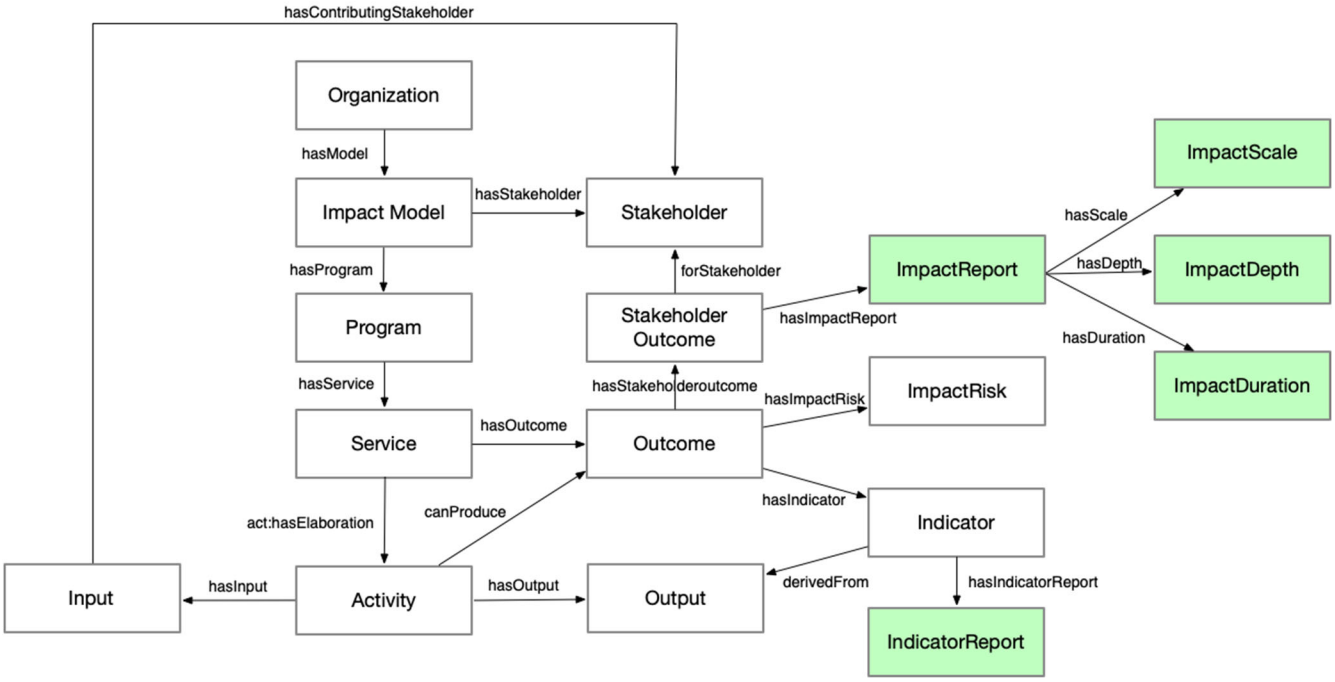

Figure 4: Detail of the impact representation specified in the Common Impact Data Standard. The Common Impact Data Standard represents the fifteen data categories defined by the Impact Management Project as well as the components of impact models including activity, output, and outcome.

The Core Ontology classes are defined in terms of the Foundation Ontology which come together to form the Common Impact Data Standard. The ontology is defined in Description Logic and published using the Semantic Web ontology language OWL.

2.1. ImpactModel

The Common Impact Data Standard is designed to represent the different variations of impact models in use by Social Purpose Organizations. The ImpactModel class is the root of a taxonomy of impact models. In this section we elaborate three: Logic Model, Outcome Chain, and Impact Measurement. The properties of each reflect the differences in focus and levels of detail.

Note that in the descriptions that follow, we use the word “key”. By “key” we mean a subset of each that are important to depict at this level of abstraction. Many impact measurement approaches use the term *material* instead of key. At present the Common Impact Data Standard does not have a class or property for materiality. Until that feature is added to the standard, we use the term “key”.

ImpactModel provides the most basic properties:

- sch:name: a string containing a title for the model.
- sch:description: a string containing a description of the model.
- sch:dateCreated: a xsd:date the model was created, in the format of yyyy-mm-dd.
- forOrganization: an object property that links to the Organization the model is for.

Class	Property	Value Restriction
ImpactModel	sch:name	exactly 1 xsd:string
	sch:description	exactly 1 xsd:string
	sch:dateCreated	exactly 1 xsd:date
	forOrganization	exactly 1 Organization

LogicModel is a subclass of ImpactModel. The top-level definition of a LogicModel contains the following properties:

- hasProgram: Identifies all the Programs being modeled.
- hasStakeholder: Identifies key stakeholders participating in the model.
- hasOutcome: Identifies key Outcomes for the model.
- hasStakeholderOutcome: identifies the outcome specific to a stakeholder.
- hasInput: Identifies key Inputs for the model.
- hasOutput: Identifies key Outputs for the model.
- hasActivity: Identifies key Activities of the model.
- act:hasResource: Identifies key Resources of the model.

Class	Property	Value Restriction
LogicModel	rdfs:subClassOf	ImpactModel
	hasProgram	only Program
	hasStakeholder	only Stakeholder
	hasOutcome	only Outcome
	hasStakeholderOutcome	only StakeholderOutcome
	hasInput	only Input
	hasOutput	only Output
	hasActivity	only Activity
	act:hasResource	only act:Resource

OutcomeChain is a subclass of ImpactModel. The top-level definition of an OutcomeChain contains the following properties:

- hasOutcome: Identifies key Outcomes for the model.
- hasActivity: Identifies key Activities of the model.
- hasIndicator: Identifies the key Indicators of the model.

Class	Property	Value Restriction
OutcomeChain	rdfs:subClassOf	ImpactModel
	hasOutcome	only Outcome
	hasActivity	only Activity
	hasIndicator	only Indicator

ImpactMeasurement is a subclass of ImpactModel it can be used to extend any of the above Impact Models to represent Impact as defined by the Impact Management Project. The top-level definition of an ImpactMeasurement contains the following properties:

- hasStakeholder: Identifies stakeholders which are the focus of the outcomes – corresponds to the “Who” in Impact Measurement.
- hasOutcome: Identifies key Outcomes for the model – corresponds to the “What” in Impact Measurement.
- hasStakeholderOutcome: identifies the importance of the outcome to the stakeholder.
- hasImpactReport: Identifies key Impacts for the model.
- hasImpactRisk: Identifies key Risks for the model.
- hasIndicator: Identifies key Indicators for the model.

Class	Property	Value Restriction
ImpactMeasurement	rdfs:subClassOf	ImpactModel
	hasStakeholder	only Stakeholder
	hasOutcome	only Outcome
	hasStakeholderOutcome	only StakeholderOutcome
	hasImpactRisk	only ImpactRisk
	hasIndicator	only Indicator
	hasImpactReport	Only ImpactReport

2.2. Organization Profile

The cids:Organization class is defined as a rfds:subClassOf org:Organization which provides the basic properties related to organization structure and behavior. cids:Organization extends org:Organization with the following properties:

- hasImpactModel: Links the organization to the one or more ImpactModels.
- useOfFunds: Is a string that describes how the funds the organization receives are used.
- sch:description: a string that describes the Organization

- sch:dateCreated: the date the Organization's record was created.

The Charity Number or Business Number of the cids:Organization is specified using the org:hasID property as defined in org:Organization below.

Class	Property	Value Restriction
Organization	rdfs:subClassOf	org:Organization
	useOfFunds	only xsd:string
	hasImpactModel	only ImpactModel
	sch:description	exactly 1 xsd:string
	sch:dateCreated	exactly 1 xsd:dateTime

Table 4: Properties of Organization Class

cids:Organization reuses and extends the TOVE Organization Ontology (Fox et al., 1998). The TOVE Organization Ontology is accessible at https://ontology.eil.utoronto.ca/tove/organization.owl. The following properties of org:Organization are inherited by cids:Organization:

- org:hasID: specifies identifiers for the Organization. They may be a business number, charity number, etc.
- ic:has Address: The main address of the Organization.
- org:hasLegalName: A string that specifies the legal name of the Organization
- org:hasLegalStatus: An instance of a class that specifies the legal status of the Organization. The legal status will differ based on country.
- ic:hasTelephone: Main phone numbers of the Organization
- org:numberOfEmployees: A non-negative integer that specifies the Organization's number of employees
- hasContact: identifies one or more people who are the contact for the Organization.
- org:consistsOf: An org:Organization may be divided into org:Divisions.

Class	Property	Value Restriction
org:Organization	org:hasID	only org:OrganizationID
	ic:hasAddress	only ic:Address
	org:hasLegalName	exactly 1 xsd:string
	org:hasLegalStatus	only owl:Thing
	ic:hasTelephone	only ic:PhoneNumber
	org:numberOfEmployees	exactly 1 xsd:nonNegativeNumber
	org:consistsOf	only org:Division
	org:hasContact	only Person
org:OrganizationID	org:issuedBy	org:Organization
	sch:identifier	exactly 1 xsd:string
	sch:dateCreated	exactly 1 xsd:dateTime

2.3. Outcome, StakeholderOutcome, and ImpactReport

As defined by the Impact Measurement Project, Outcomes "are what stakeholders experience as a result of an enterprise's activities. They can be positive or negative, intended or unintended." CIDS captures the key properties of an outcome by representing:

- hasStakeholderOutcome: Identifies the impact it has on Stakeholders.
- forDomain: Identifies the Domains that the Outcome aligns with.
- hasIndicator: Identifies the set of Indicators the organization assigns to the Outcome.
- canEnable: Links an Outcome to a Service or Activity that is made possible due to the result of the Outcome. It abstracts a more detailed specification of Activities producing States that enable other activities. It is used primarily for representing Outcome Chains.
- canProduce: Links an Outcome to another Outcome. It abstracts the underlying activity chain that usually links one Outcome to another. It is used primarily for representing Outcome Chains.
- i72:located_in: Identifies the spatial location the outcome is define for.
- hasImpactRisk: Identifies the impact risks associated with the outcome.
- oep:partOf: Identifies the Impact Model it is a component of.

Domain identifies a sector that the Outcome applies to. It specifies the Organization that is the source of the Domain codes (i.e., codelist).

- forOrganization: The Organization that defined the Domain.
- hasSpecification: The URI where the codelist can be found.
- sch:description: A description of the source of the Domain codelist.
- sch:codeValue: The code for the specific domain.

Class	Property	Value Restriction
Outcome	hasStakholderOutcome	only Stakeholder Outcome
	forDomain	only Domain
	hasIndicator	only Indicator
	canEnable	only (Service or Activity)
	canProduce	only Outcome
	i72:located_in	only i72:Feature
	hasImpactRisk	only ImpactRisk
	oep:partOf	exactly 1 ImpactModel
	sch:name	exactly 1 xsd:string
	sch:description	exactly 1 xsd:string
	sch:dateCreated	exactly 1 xsd:dateTime
Domain	forOrganization	exactly 1 Organization
	hasSpecification	exactly 1 xsd:anyURI
	sch:description	exactly 1 xsd:string
	sch:codeValue	exactly 1 xsd:string
UNSDGDomain (example)	rdfs:subClassOf	Domain
	forOrganization	value unitedNations

	hasSpecification	value https://sdgs.un.org/goals
	sch:description	value “United Nations Sustainable Development goals”

The StakeholderOutcome specifies for a specific Stakeholder whether the Outcome will have a positive, negative, or neutral affect on the Stakeholder, and the nature of the impact:

- forStakeholder: Identifies the Stakeholder affected.
- fromPerspectiveOf: Identifies the Stakeholder who is determining the importance of the Impact.
- hasImportance: Specifies the nature of the importance. One of {“high importance”, “moderate important”, “neutral”, “unimportant”}.
- isUnderserved: A Boolean that denotes if the stakeholder underserved in relation to your a specific outcome.
- intendedImpact: Identifies the intended direction of the change – note that ImpactReport captures the actual direction.
- hasIndicator: Identifies the set of Indicators the Organization assigns to the Outcome but are specific to this Stakeholder.
- hasImpactReport: identifies the set of ImpactReport that report on the results pertaining to each Outcome.

Class	Property	Value Restriction
StakeholderOutcome	forStakeholder	exactly 1 Stakeholder
	fromPerspectiveOf	exactly 1 Stakeholder
	hasImportance	exactly 1 {“high importance”, “moderate important”, “neutral”, “unimportant”}
	isUnderserved	exactly 1 xsd:boolean
	intendedImpact	exactly 1 {“positive”, “negative”, “neutral”}
	hasIndicator	only Indicator
	hasImpactReport	only ImpactReport

ImpactReport specifies the “How Much” dimension of the Impact Management Project. It reports on scale, depth, and duration of the Outcome.

- forOrganization: Links to the Organization for whom this report is associated.
- forOutcome: Links to the Organization’s Outcome for whom this report is associated.
- hasReportedImpact: Specifies one of three values for the impact as measured – positive, negative or neutral.
- hasImpactScale: Specifies the number of stakeholders who experience the outcome.
- hasImpactDepth: Impact Depth is the degree of difference between the assumed condition that would take place without intervention and the condition with interventions implemented on the stakeholders. Impact Depth can be calculated using the measured or estimated counterfactual.
- hasImpactDuration: Impact Duration is the length of time that a stakeholder experiences an impact from the initial implementation.
- hasExpectation: A string field where any expectations that the organization has can be provided.
- hasComment: A string field where any comments that the organization has can be provided.

ImpactScale, ImpactDepth and ImpactDuration have the following properties:

- hasIndicator: The Indicator used to measure the Impact.
- prov:wasGeneratedBy: A string that can used to describe the method used to determine the impact.
- sch:description: A string that can used to provide additional information about the impact.
- i72:value: Links to the value of the impact specified as an i72:Measure
- hasCounterfactual: Links to a Counterfactual in which the value of the counterfactual is specified as a i72:Quantity, and the source of the counterfactual is specified in a string.

Class	Property	Value Restriction
ImpactReport	forOrganization	exactly 1 Organization
	forOutcome	exactly 1 Outcome
	hasReportedImpact	exactly 1 {“positive”, “negative”, “neutral”}
	hasImpactScale	exactly 1 ImpactScale
	hasImpactDepth	exactly 1 ImpactDepth
	hasImpactDuration	exactly 1 ImpactDuration
	hasExpectation	exactly 1 xsd:string
	hasComment	exactly 1 xsd:string
ImpactScale	hasIndicator	exactly Indicator
	prov:wasGeneratedBy	exactly 1 (sur:Survey or Interview or Estimate or Computation)
	sch:description	exactly 1 xsd:string
	i72:value	exactly 1 i72:Measure
	hasCounterfactual	only Counterfactual
ImpactDepth	hasIndicator	exactly Indicator
	prov:wasGeneratedBy	exactly 1 (sur:Survey or Interview or Estimate or Computation)
	sch:description	exactly 1 xsd:string
	i72:value	exactly 1 i72: Measure
	hasCounterfactual	only Counterfactual
ImpactDuration	hasIndicator	exactly Indicator
	hasStartDate	exactly 1 xsd:dateTime
	hasEndDate	exactly 1 xsd:dateTime
	prov:wasGeneratedBy	exactly 1 (sur:Survey or Interview or Estimate or Computation)
	sch:description	exactly 1 xsd:string
	i72:value	exactly 1 i72: Measure
	hasCounterfactual	only Counterfactual
Counterfactual	i72:located_in	only i72:Feature
	i72:for_time_interval	exactly 1 time:DateTImeInterval
	prov:wasGeneratedBy	exactly 1 (sur:Survey or Interview or Estimate or Computation)
	sch:description	exactly 1 xsd:string
	i72:value	i72: Measure

2.4. Stakeholder and Stakeholder Characteristics

The Stakeholder class is a subclass of an Organization or Person. It identifies what activities they perform using the *performs* property, where they are located geographically using the i72:*located_In* property. Some methods for measuring impact, such as a Social Return on Investment, require that Social Purpose Organizations distinguish between the beneficiary and contributing stakeholders. A BeneficiaryStakeholder is a stakeholder that benefits from a logic model's outcome. A ContributorStakeholder is a stakeholder that contributes input to ensure a service can produce outcomes. These are explicitly defined as separate classes to highlight the fact that a contributor stakeholder is not always the same instance as the beneficiary stakeholder. The distinction is available in the ontology but is not required.

Finally, we add the five types of Stakeholders. As a starting point we have identified Stakeholders specified by the Impact Management Project: Customers, Employees, Communities, Suppliers, and Planet, as well as a distinction between Contributing and Beneficial stakeholders as suggested by Social Value International.

Stakeholder contains the following properties:

- sch:name (title): A title for the stakeholder as a string.
- sch:description: A general description of the stakeholder as a string.
- hasCatchmentArea: Specifies the regional span of the stakeholders.
- hasStakeholderCharacteristic: Specifies characteristics of the stakeholder
- performs: Links to the activities performed by the stakeholder.
- i72:located_in: Links to the specific geographic area in which the stakeholder is located.
- oep:part of: Links the Impact Model that the stakeholder is being specified for.

Class	Property	Value Restriction
Stakeholder	rdfs:subClassOf	(Organization or cids:Person)
	sch:name (title)	exactly 1 xsd:string
	sch:description	exactly 1 xsd:string
	hasCatchmentArea	exactly 1 {"local", "provincial", "national", "multinational", "global" }
	hasStakeholderCharacteristic	only StakeholderCharacteristic
	performs	some Activity
	i72:located_in	only i72:Feature
	oep:partOf	only ImpactModel
BeneficiaryStakeholder	rdfs:subClassOf	Stakeholder
	beneiftsFrom	min 1 Outcome
	org:hasRole	only {Client, Community, Employee, Planet, Supplier}
ContributingStakeholder	rdfs:subClassOf	Stakeholder
	contributes	min 1 Input
Client	rdfs:subClassOf	Stakeholder
Employee	rdfs:subClassOf	Stakeholder
Community	rdfs:subClassOf	Stakeholder
Supplier	rdfs:subClassOf	Stakeholder
Planet	rdfs:subClassOf	Stakeholder

The StakeholderCharacteristic class is used to define a characteristic of a Stakeholder. It has the following properties:

- forOrganization: Specifies the Organization that defined the standard codes, if used.
- hasSpecification: Specifies the URI for the source of the code list of stakeholder characteristics.
- sch:identifier: Specifies the code for the characteristic.
- sch:name: Specifies the name or title of the characteristic.
- sch:description: Specifies the description of the characteristic.
- sch:codeValue: The Boolean value for whether the Stakeholder has characteristic.

Class	Property	Value Restriction
StakeholderCharacteristic	forOrganization	exactly 1 Organization
	hasSpecification	exactly 1 xsd:anyURI
	sch:identifier	exactly 1 xsd:string
	sch:name	exactly 1 xsd:string
	sch:description	exactly 1 xsd:string
	sch:codeValue	exactly 1 xsd:boolean

2.5. Indicator and IndicatorReport

In this section we define the Indicator and IndicatorReport classes. Indicator is a sub class of i72:Indicator (see section 3.5), which provides properties for units of measure, time and value. Indicator extends the definition of i72:Indicator with the following properties:

- definedBy: Links to the cids:Organization that defined the Indicator.
- forOutcome: Links to the Outcome's the Indicator measures. .
- usesOutput: Links the Outputs that the indicator uses in defining its value. This is useful when indicators are computed using output information. This property can be ignored if dcat:dataset is used. The more detailed specification of the definition and computation of the indicator is provided by the i72 Indicator properties.
- hasBaseline: Links to an i72:Measure that specifies a baseline describing the existing condition before intervention. It is used for comparison purposes. Baseline data can be measured or estimated using other datasets. It should be expressed using the same units as the indicator.
- hasThreshold: Links to an i72:Measure that specifies a minimum or maximum quantity that limits which values an indicator can assume.
- prov:wasGeneratedBy: Links to a method that specifies how the Indicator was derived.
- dcat:dataset: Links to a dcat:Dataset that specifies the data used to derive the value if the method is Computation.
- hasIndicatorReport: Links to all of the indicator reports for this indicator.
- hasAccess: Links to the organizations that can read this Indicator.
- hasIndicatorStandard: Links to an IndicatorStandard that identifies an existing standard/codelist that this Indicator is an example of.
- dqv:target: Links to quality policies that the Indicator should conform to.

Class	Property	Value Restriction
Indicator	rdfs:subClassOf	i72:Indicator
	definedBy	exactly 1 cids:Organization
	forOutcome	only Outcome
	usesOutput	only Output
	hasBaseline	exactly 1 i72:Measure
	hasThreshold	exactly 1 i72:Measure
	prov:wasGeneratedBy	exactly 1 (sur:Survey or Interview or Estimate or Computation)
	dcat:dataset	only dcat:Dataset
	hasIndicatorReport (inverse forIndicator)	only IndicatorReport
	hasAccess	only cids:Organization
	hasIndicatorStandard	only IndicatorStandard
	dqv:target	only dqv:QualityPolicy
	sch:name	exactly 1 xsd:string
	sch:description	exactly 1 xsd:string
	sch:dateCreated	exactly 1 xsd:dateTime

The IndicatorStandard class identifies the various standards/codelists that exist for a particular indicator. It supports the ability to compare an outcome to nationally or internationally recognized by means of its indicators.

- forOrganization: Specifies the Organization that defined the standard/codelist.
- sch:identifier: Specifies the standard's code number for the indicator.
- sch:name: Specifies the standard's name or title of the indicator.
- sch:description: Specifies the standard's description of the indicator.
- i72:value: Defines the threshold value from the standard.

Class	Property	Value Restriction
IndicatorStandard	forOrganization	exactly 1 cids:Organization
	sch:identifier	exactly 1 xsd:string
	sch:name	exactly 1 xsd:string
	sch:description	exactly 1 xsd:string
	i72:value	exactly 1 i72:Measure

IndicatorReport is used to report the value of an indicator for some time interval It contains the following properties:

- forOrganization: Links to the Organization that submits the report.
- forIndicator: Links to the Indicator that is being reported.
- i72:value: Links to the value specified as a i72:Measure.
- prov:wasGeneratedBy: Links to the method by which the Indicator was derived.
- dcat:dataset: Links to the dcat:Dataset that specifies the datasets used to derive the value if the method is Computation.
- i72:for_time_interval: Specifies the time interval that the Indicator Report covers.

- dqv:hasQualityAnnotation: Specifies any annotations regarding the quality of the reported indicator quantity.
- dqv:conformsTo: Specifies the standards that the Indicator being reported conforms to.
- dqv:hasQualityMeasurement: Specifies a metric that represents the evaluation of the indicator.
- hasAccess: Links to the organizations that can read this Indicator report.
- hasComment: A string property in which a general comment for the report can be specified.

Class	Property	Value Restriction
IndicatorReport	forOrganization	exactly 1 Organization
	forIndicator	exactly 1 Indicator
	i72:value	exactly 1 i72:Measure
	prov:wasGeneratedBy	exactly 1 (sur:Survey or Interview or Estimate or Computation)
	dcat:dataset	only dcat:Dataset
	i72:for_time_interval	exactly 1 time:DateTimeInterval
	dqv:hasQualityAnnotation	only dqv:QualityAnnotation
	dqv:conformsTo	only dcterms:Standard
	dqv:hasQualityMeasurement	only dqv:QualityMeasurement
	hasAccess	only Organization
	hasComment	only xsd:string
	sch:name	exactly 1 xsd:string
	sch:dateCreated	exactly 1 xsd:dateTime

2.6. ImpactRisk

ImpactRisk “assesses the likelihood that impact will be different than expected, and that the difference will be material from the perspective of people or the planet who experience impact.” Stating the riskiness of the impact is important for interpreting the subsequent results. The Impact Management Project recommends that as part of any impact assessment, the risk of the impact be considered as one of the five dimensions of performance.

The following defines the taxonomy of risk and key properties:

- forOutcome: Identifies the Outcome the risk is associated with.
- hasLikelihood: Identifies the likelihood that the risk will occur.
- hasConsequence: Identifies the degree of impact the risk could have.
- hasMitigation: A string that specifies a mitigation plan or references a document.

Note that the subclasses of risk do not have properties that distinguish one from another. These would be provided in later versions, as needed.

Class	Property	Value Restriction
ImpactRisk	forOutcome	only Outcome
	hasLikelihood	exactly 1 {veryUnlikely , unlikely, likely, veryLikely, lowRisk, mediumRisk, highRisk}

	hasConsequence	exactly 1 {minimal, average, severe}
	hasMitigation	exactly 1 xsd:string
	sch:description	exactly 1 xsd:string
	sch:identifier	exactly 1 xsd:string
EvidenceRisk	rdfs:subClassOf	Only ImpactRisk
ExternalRisk	rdfs:subClassOf	Only ImpactRisk
StakeholderParticipationRisk	rdfs:subClassOf	Only ImpactRisk
DropOffRisk	rdfs:subClassOf	Only ImpactRisk
EfficiencyRisk	rdfs:subClassOf	Only ImpactRisk
ExecutionRisk	rdfs:subClassOf	Only ImpactRisk
AlignmentRisk	rdfs:subClassOf	Only ImpactRisk
EnduranceRisk	rdfs:subClassOf	Only ImpactRisk
UnexpectedImpactRisk	rdfs:subClassOf	Only ImpactRisk

2.7. Program

A program defines a set of services that focus on a shared set of Outcomes. For example, a “poverty reduction program” can be made up of a set of Services such as mobiles services that provides food and clothing to those that live on the street, and a training service that provides basic skills for those living on the street. A Program has a set of Stakeholders that it may contribute or benefit:

- hasService: Identifies the Services that make up the Program.
- hasOutcome: Identifies the Outcomes that the program is trying to achieve.
- hasContributingStakeholder: Identifies the stakeholders that contribute to the Program.
- hasBeneficialStakeholder: Identifies the stakeholders that benefit from the Program.
- hasInput: Identifies the Inputs to the Program.
- hasOutput: Identifies the Outputs of the Program.

Class	Property	Value Restriction
Program	rdfs:subClassOf	Activity
	sch:name	exactly 1 xsd:string
	sch:description	exactly 1 xsd:string
	hasService	only Service
	hasOutcome	only Outcome
	hasContributingStakeholder	only ContributingStakeholder
	hasBeneficialStakeholder	only BeneficialStakeholder
	hasInput	only Input
	hasOutput	only Output

2.8. Service

A Program is composed of one or more Services. As described in the Program description, a poverty reduction program can have many services with each service comprised of different activities, Inputs, Outputs and Outcomes.

- act:hasSubActivity: Identifies the Activities that make that comprise the Service.
- hasInput: Identifies the Inputs to the Service.
- hasOutput: Identifies the Outputs of the Service.
- hasOutcome: Identifies the Outcomes that are specific to the Service.
- hasContributingStakeholder: Identifies the stakeholders that contribute to the Service.
- hasBeneficialStakeholder: Identifies the stakeholders that benefit from the Service.
- beneficiarySizeStart: Number of beneficial stakeholders at the beginning of the service time interval.
- beneficiarySizeEnd: Number of beneficial stakeholders at the end of the service time interval.
- i72:for_time_interval: Time interval over which the service is provided.

Class	Property	Value Restriction
Service	rdfs:subClassOf	Activity
	oep:partOf	exactly 1 ImpactModel
	sch:name	exactly 1 xsd:string
	sch:description	exactly 1 xsd:string
	act:hasSubActivity	only Activity
	hasInput	only Input
	hasOutput	only Output
	hasOutcome	only Outcome
	hasContributingStakeholder	only ContributingStakeholder
	hasBeneficialStakeholder	only BeneficialStakeholder
	beneficiarySizeStart	exactly 1 xsd:positiveInteger
	beneficiarySizeEnd	exactly 1 xsd:positiveInteger
	i72:for_time_interval	exactly 1 time:DateTImeInterval

2.9. Activity

Activity defines the “actions” performed by an organization to implement a Service. CIDS’s Activity class is defined to be a subclass of the TOVE Activity, and is extended by including properties for Input, Output and what Service or Activity it is a subActivityOf. An activity’s type is based on its outcome rather than service or input. This allows activities to be classified by the type of change they produce rather by what resources they use (input) or who performs the activity (service). Its properties are:

- canProduce: Specifies the Outcome that results from performance of the Activity. It is used primarily for representing Outcome Chains.
- hasInput: Specifies the Input to the Activity.
- hasOutput: Specifies the Output of the Activity.

- hasCode: Specifies zero or more codes, created by various organizations, to identify a type of Activity, e.g., ICHI – International Classification of Health Interventions activities.
- act:subActivityOf: Specifies the Service or Activity that this Activity is part of.

An instance of an ActivityCode is used to specify a particular code created by a standards organization. Its properties are:

- forOrganization: Specifies the Organization that created the code, e.g., ICHI, ISO, and so on.
- sch:identifier: Specifies the code number.
- sch:name: Specifies the name or title of the code.
- sch:description: Specifies the description of the code.

Class	Property	Value Restriction
Activity	rdfs:subClassOf	act:Activity
	canProduce	only Outcome
	oep:partOf	exactly 1 ImpactModel
	hasInput	only Input
	hasOutput	only Output
	hasCode	only ActivityCode
	act:subActivityOf	only (Service or Activity)
ActivityCode	forOrganization	exactly 1 Organization
	sch:identifier	exactly 1 xsd:string
	sch:name	exactly 1 xsd:string
	sch:description	exactly 1 xsd:string

2.10. Input

A key component of impact models are Inputs. Inputs specify the resources required by a Social Purpose Organization to produce results (Ralser, 2008). An Input is provided by a contributing stakeholder and may come in many forms. We identify three broad categories of Input:

- FinancialInput represents a monetary resource, with a monetary unit of measure, such as donating cash or paying off debt.
- SkillInput is any type of skills-based expertise such as legal, translation, carpentry, etc.
- PhysicalInput is any type of physical item, such as food, clothing, furniture, etc.

Properties common across all types of Input are:

- eof:partOf: Specifies the impact models this Input is part of.
- inputFor: Identifies the Program, Service or Activity this is an input for.
- hasContributingStakeholder: Identifies the stakeholder that contributes the resource.
- hasType: Specifies the type of Resource by denoting the relevant subclass of Resource.
- hasPlannedAmount: Specifies the Quantity of Input (which in turn specifies the unit of measure) planned to be used by the activity.
- hasActualAmount: Specifies the Quantity of Input that was used (which in turn specifies the unit of measure) used.

- i72:for_time_interval: Specifies the time interval over which the Input is provided.

Class	Property	Value Restriction
Input	eof:partOf	exactly 1 ImpactModel
	inputFor	only (Program or Service or Activity)
	hasContributingStakeholder	only ContributingStakeholder
	hasType	exactly 1 act:Resource
	hasPlannedAmount	exactly 1 i72:Measure
	hasActualAmount	exactly 1 i72:Measure
	i72:for_time_interval	only time:DateTimeInterval
	sch:name	exactly 1 xsd:string
	sch:description	exactly 1 xsd:string
FinancialInput	rdfs:subClassOf	Input
	hasType	only FinancialResource
	hasAmount	exactly 1 (i72:Quantity and i72:unit_of_measure i72:Monetary_unit)
SkillInput	rdfs:subClassOf	Input
	hasType	only SkillResource
PhysicalInput	rdfs:subClassOf	Input
	hasType	only PhysicalResource

2.11. Output

Outputs are a quantitative summary of an activity. For example, if the activity is ‘we provide training’ and the output is ‘we trained 50 people to NVQ level 3’ (CED, 2012). Or a production output could produce 100 meals for the homeless. Basic to these outputs is “what” has been produced and the quantity.

- eof:partOf: Specifies the impact models this Output is part of.
- forActivity: Identifies the Activity or Service that produces the Output.
- i72:value: Identifies that amount that is produced.
- produces: Identifies the Resource that is produced such as a skill, or a type of Meal.
- usedByIndicator: Identifies the Indicators that use this Output in determining the value of the Indicator.

Class	Property	Value Restriction
Output	oep:partOf	Exactly 1 ImpactModel
	forActivity	only (Service or Activity)
	i72:value	exactly 1 i72:Measure
	produces	exactly 1 act:Resource
	usedByIndicator	only Indicator
	sch:name	exactly 1 xsd:string
	sch:description	exactly 1 xsd:string

3. Foundation Ontology: Common Impact Data Standard

The Foundation Ontology is a set of pre-existing ontologies, that are used by the Core Ontology. They provide basic representations of time, address & phone number, measurement, indicator, person and activity.

3.1. Date/Time – OWL-Time:2020

3.1.1. Introduction

Time is both absolute and relative. To understand impact data, it is often necessary to know at what time something occurred, and also whether something occurred before, after or during some other event. It is not enough to record a date. An impact ontology requires a much richer representation of time that supports reasoning about time points, time intervals and the relationships among them. 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 Candidate Recommendation 26 March 2020” accessed at https://www.w3.org/TR/owl-time/.

3.1.2. Core Classes and Properties

This section summarizes some of the core classes and properties defined in OWL-Time. Fundamental to any conceptual model, including an impact 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 newcomer 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. This means that the beginning time is before the ending 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. [i]

Figure 5 depicts the core classes that comprise OWL-Time.

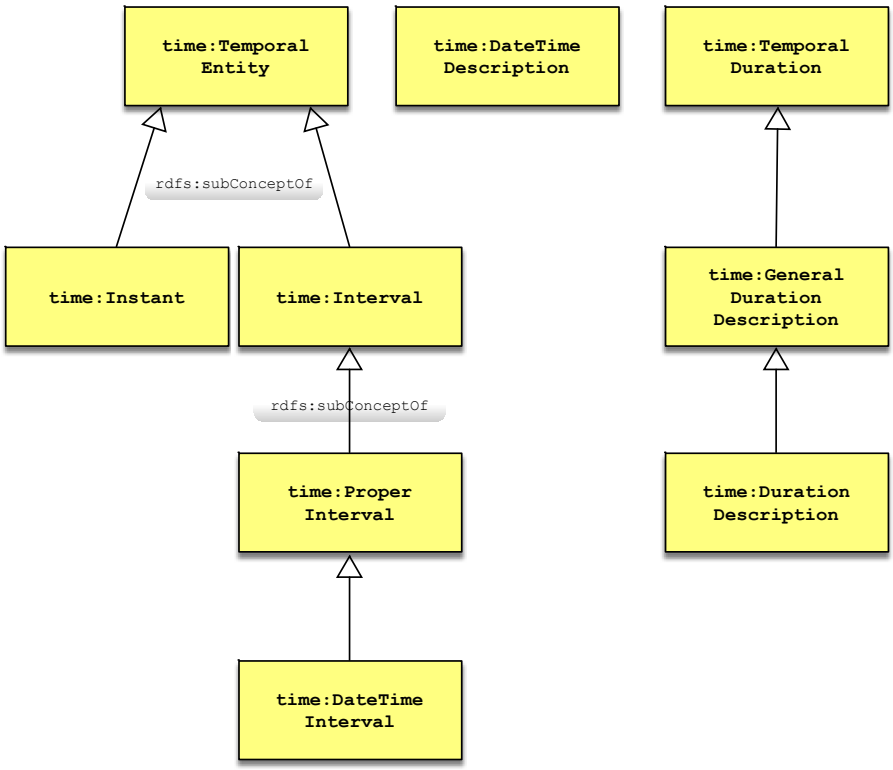

Figure 5: Time Concepts

Figure 6 depicts the relationships that TemporalEntity has with the other classes.

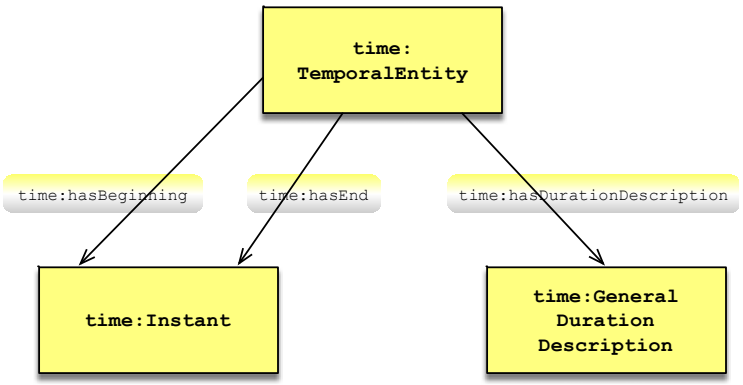

Figure 6: TemporalEntity Relationships

3.1.3. Formal Specification

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

Class	Property	Value Restriction
time:TemporalEntity	rdfs:subClassOf	time:TemporalThing
	time:hasBeginning	only 1 time:Instant
	time:hasEnd	only 1 time:Instant
	time:hasDurationDescription	only 1 time:DurationDescription
time:Instant	rdfs:subClassOf	time:TemporalEntity
	time:inDateTime	exactly 1 time:DateTimeDescription
	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 time: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)

3.2. Address – tove/icontact

3.2.1. Introduction

Addresses across the globe vary in the types of information they include. For example, a UK or Indian address may refer to a building name and a section of a city. This document reuses the International Contact Ontology which is equipped to represent most global addresses. The International Contact Ontology accessed can be accessed at http://ontology.eil.utoronto.ca/tove/icontact.owl.

3.2.2. Formal Specification

The concept of Address class deconstructs the address into its constituents.

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	{main, division, regional, branch}
ic:StreetDirection	owl:equivalentTo	{east, north, south, west}
ic:StreetType	owl:equivalentTo	{avenue, boulevard, circle, crescent, drive, road, street}

3.3. Phone Number – tove/icontact

The PhoneNumber class decomposes a phone number into its individual parts. Properties in blue are inherited from the superclass ic:PhoneNumber. This document reuses the International Contact Ontology.

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	{mainline, faxPhone, cellPhone}

3.4. Measurement – ISO/IEC 21972:2020

3.4.1. 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 those used in indicator reports, are of the same type. For example, to clarify if “5” is a count or a percentage; or to ensure that when two numbers are used in a ratio, that are expressed in the same scale, for example, hundreds vs thousands. Consider an Indigenous advocacy group that is working to end indigenous homeless. They use data from two different sources to track the percentage of the homeless that are Indigenous. To do this, they need to know that the data of homeless Indigenous and total homeless are of the same scale, for example, hundreds vs thousands.

This document reuses “ISO/IEC 21972:2020 Information technology — Upper level ontology for smart city indicators” (accessible at https://www.iso.org/standard/72325.html) which provides a representation for representing the definition of indicators.

3.4.2. Core Classes and Properties

The CIDS representation of measurement concepts reuses ISO 21972 which is based on the OM measurement ontology (Rijgersberg et al., 2013). The top row of Figure 7 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 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'.

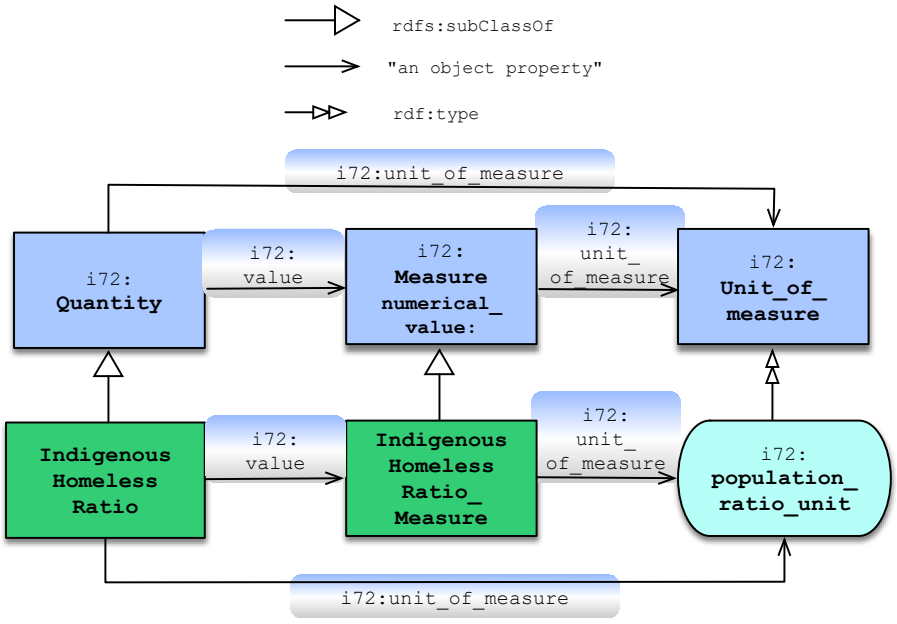

Figure 7: Basic Measurement Classes

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 as outlined below and illustrated in Figure 8:

- 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).

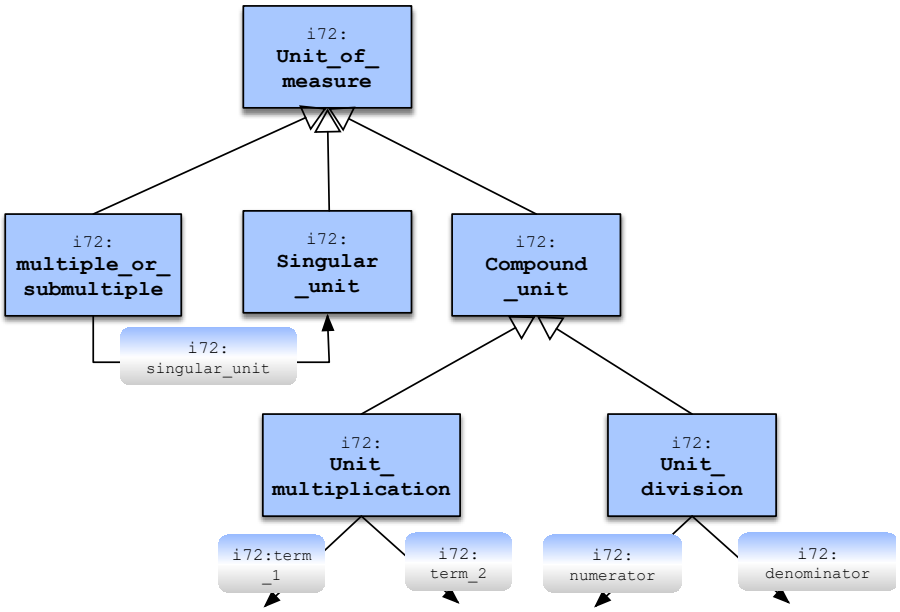

Figure 8: Unit of Measure Taxonomy

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[ii], a zero point can be defined. The measurement scale taxonomy is illustrated in Figure 9.

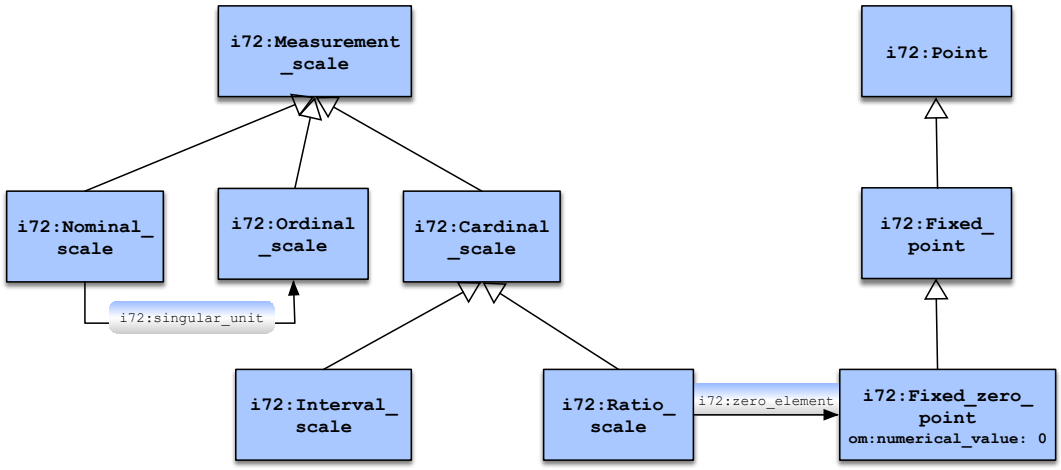

Figure 9: Measurement Scale Taxonomy

3.4.3. 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_submultiple	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)”.

3.5. Indicator – ISO/IEC 21972:2020

3.5.1. 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 verifiable. 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[iii]. In this section we present the core concepts for defining indicators.

This document reuses “ISO/IEC 21972:2020 Information technology — Upper level ontology for smart city indicators” (accessible at https://www.iso.org/standard/72325.html).

3.5.2. Core Classes and Properties

The CIDS representation of indicator measurement concepts reuse 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 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 a unit of measure of the size of a population. It is defined to be an individual of a Cardinality_unit that is a subclass of a Singular_unit. Figure 10 depicts the specification of the Cardinality_unit.

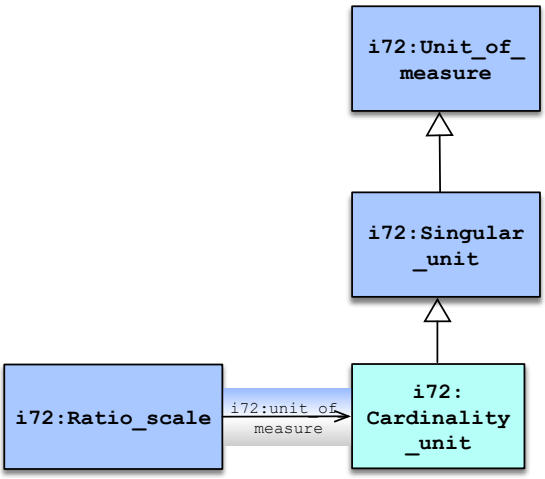

Figure 10: Cardinality_unit Definition

In Figure 10, 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.

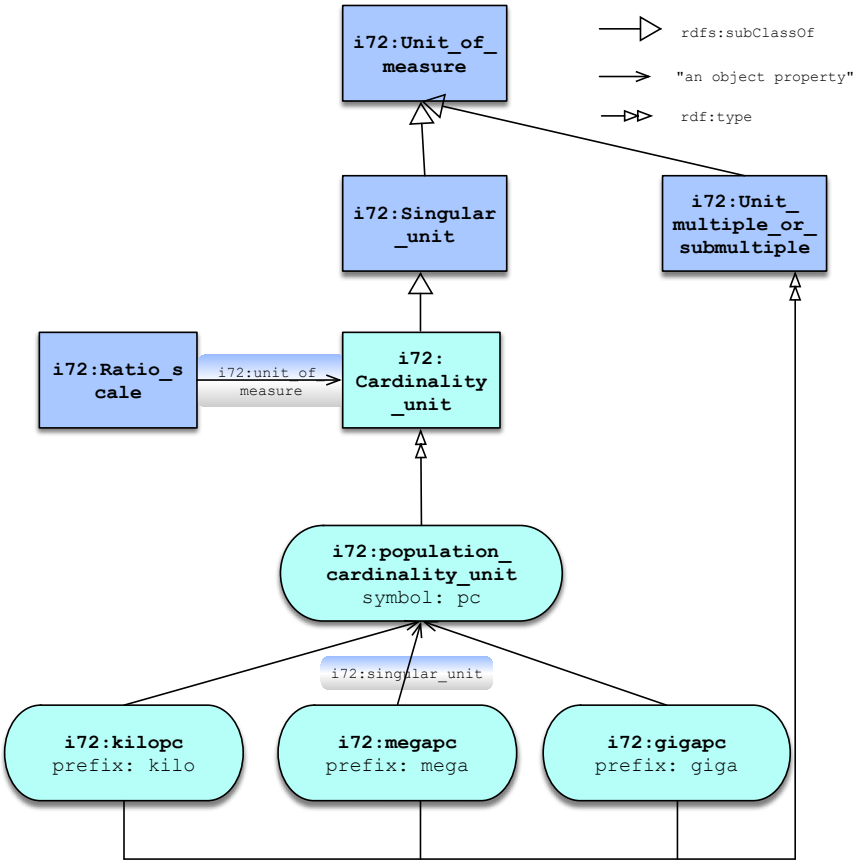

Figure 11: population_cardinality_unit Definition

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 (see Figure 11). 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 12 provides the unit of measures for populations (pc) and population ratios (pc/pc).

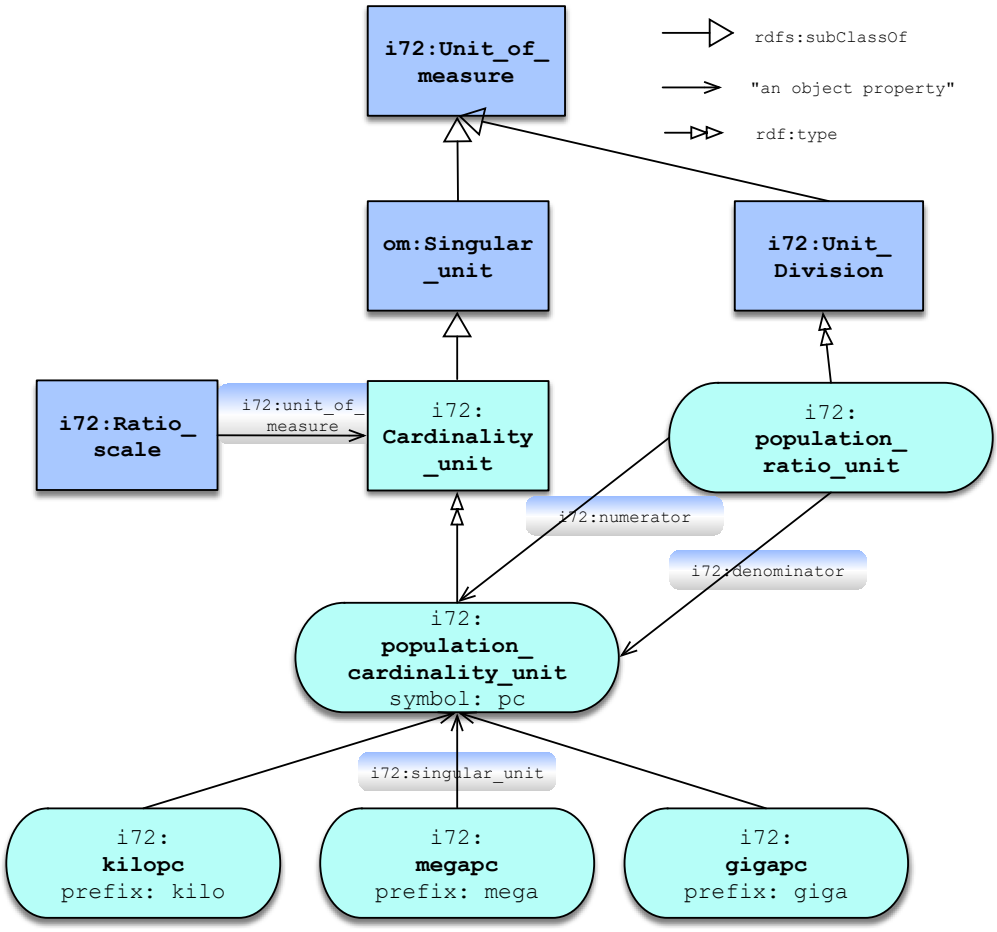

Figure 12: Depicts the population ratio unit definition

3.5.3. Formal Specification

In addition to the classes and properties identified in section 2.4, the following table specifies the key concepts for representing indicator definitions (using the Manchester syntax for Description Logic Full).

Class	Property	Value Restriction
i72:Indicator	rdfs:subClassOf	i72:Quantity
	i72:unit_of_measure	exactly 1 i72:Unit_of_measure
	i72:value	exactly 1 i72:Measure
	i72:for_time_interval	exactly 1 time:DateTImeInterval
i72:RatioIndicator	rdfs:subClassOf	i72:Indicator
	i72:unit_of_measure	exactly 1 i72:Unit_division
	i72:numerator	exactly 1 i72:Quantity
	I72:deominator	Exact;u 1 i72:Quantity

The basic definitions for population cardinality are as follows:

Class	Property	Value Restriction
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
Individual	Property	Value
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.

Figure 13 depicts the translation of the Indicator “Average Number of Skills each Job Seeker gained”, the indicator ontology, which is based on ISO 21972.

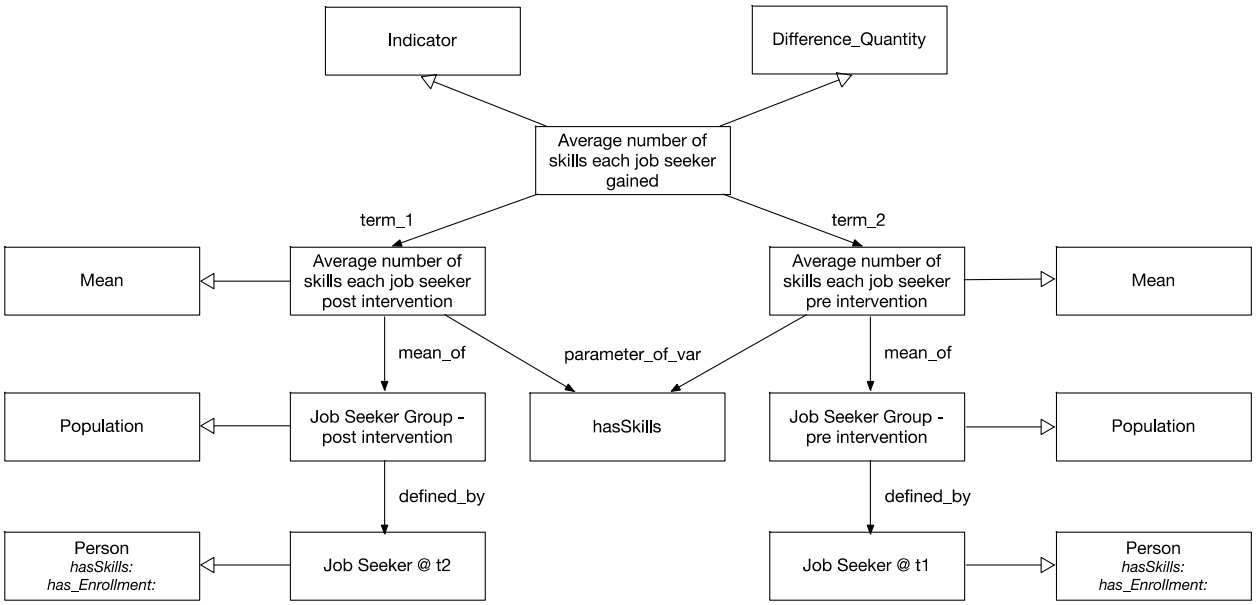

Figure 13: Indicator Pattern

3.6. Person

3.6.1. 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 software company that is using the data ontology.

This document reuses and extends the “CWRC Ontology Specification” because it has already defined many objects that are relevant to CAFO. CWRC Ontology Specification is accessible at http://sparql.cwrc.ca/ontology/cwrc.html.

3.6.2. Core Concepts 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.

3.6.3. Formal Specification

Class	Property	Value Restriction
Person	rdfs:subClassOf	sch:Person
	ic:hasAddress	only ic:Address
	ic:hasPhoneNumber	only ic:PhoneNumber
	ic:hasEmail	only xsd:string

	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
	cwrd:parentOf	Only Person
	csrc:hasGender	only cwrc:Gender
	csrc:hasEthnicity	only cwrc:Ethnicity
	csrc:hasReligion	only cwrc:Religion
	hasOccupation	only Occupation
	org:plays	only org:Role
	rel:spouseOf	only Person
	hasDisability	only Disability
	hasDisease	only Disease
	hasImmigrationStatus	only ImmigrationStatus
	hasMaritalStatus	some MaritalStatus
	sch:indentifier	only xsd:string

3.7. Activity – tove/activity

3.7.1. Introduction

In order to represent an organization's impact model, it is often necessary to represent the activities that the organization undertakes to affect change.

This document reuses and extends the "TOVE Activity ontology" (Fox et al., 1993). TOVE Activity ontology is accessible at http://ontology.eil.utoronto.ca/tove/activity.owl.

3.7.2. Core Classes and Properties

Action is represented by the combination of an activity and its corresponding enabling and caused states (Figure 12). 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.

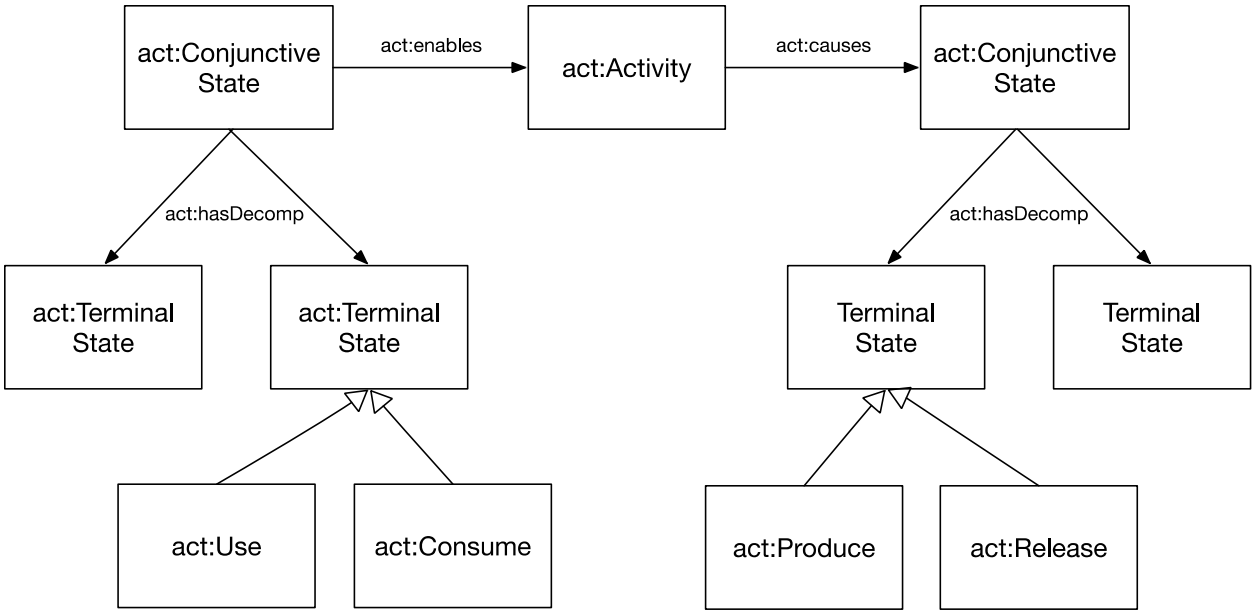

Figure 12: Activity-State Cluster

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 executed 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 act: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:** Specifies that all substates / at least one substate must be satisfied.

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**: A 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.

3.7.3. 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:hasStatus	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:hasResource	only act:Resource
act:NonTerminalState	rdfs:subClassOf	act:State
	owl:disjointWtih	act:TerminalState
	act:hasSubState	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
act:Resource	rdfs:subClassOf	owl:Thing

4. Example Use

In this section we show how the Common Approach Core Ontology can be used as an interchange format among SPOs, Impact Measurement platform providers and other repositories. We demonstrate an cids:Outcome using two formats: JSON-LD and Turtle.

4.1. cids:Outcome in JSON-LD

The following depicts a cids:Outcome in JSON-LD[1], the linked data version of JSON. @context defines the prefixes to be used in the body of the json. @type attribute specifies that the JSON object is a cids:Outcome All attribute names correspond to the properties defined for a cids:Organization. Note that some of the properties have IRI's for values, and others have literals.

```
{   "@context" : {
        "cids" : "http://ontology.eil.utoronto.ca/CAO/cids#" ,
        "uwgt" : "https://www.unitedwaygt.org/LD#"
    } ,
    "@type" : "cids:Outcome" ,
    "cids:hasStakeholderOutcome" : [ "uwgt:impact1", "uwgt:impact2" ] ,
    "cids:forDomain" : "cids:sdg1" ,
    "cids:hasIndicator" : [ "uwgt:ind1", "uwgt:ind2", "uwgt:ind3" ]
}
```

4.2. Cids:Outcome in Turtle

The following depicts a cids:Outcome in Turtle[2], a serialization format for RDF. All attribute names correspond to the attributes defined for a cids:Outcome.

```
@prefix          cids : <http://ontology.eil.utoronto.ca/CAO/cids#> .
@prefix          uwgt : <https://www.unitedwaygt.org/LD#> .
uwgt:outcome1  rdf:type  cids:Outcome ;
          cids:hasStakeholderOutcome  uwgt:impact1, uwgt:impact2 ;
          cids:forDomain  caco:sdg1;
          cids:hasIndicator  uwgt:ind1, uwgt:ind2, uwgt:ind3 .
```

4.3. cids:Organization in JSON-LD

The following depicts a cids:Organization in JSON-LD[3], the linked data version of JSON. @context defines the prefixes to be used in the body of the json. @type attribute specifies that the JSON object is a cids:Organization.

[1] https://www.w3.org/TR/json-ld11/
[2] https://www.w3.org/TR/turtle/
[3] https://www.w3.org/TR/json-ld11/

All attribute names correspond to the properties defined for a cids:Organization. Note that some of the properties have IRI's for values, and others have literals.

```
{   "@context" : {
        "caco" : "http://ontology.eil.utoronto.ca/CAO/caco#" ,
        "cafo" : "http://ontology.eil.utoronto.ca/CAO/cafo#" ,
        "sch" : "http://schema.org/" ,
        "org" : "http://ontology.eil.utoronto.ca/organization#" ,
        "ic" : "http://ontology.eil.utoronto.ca/icontact#" ,
        "uwgt" : "https://www.unitedwaygt.org/LD#" ,
    }
    "@type" : "cafo:Organization" ,
    "org:hasID" : "https://www.unitedwaygt.org/" ,
    "ic:has Address" : "uwgt:uwgt_address" ,
    "ord:hasLegalStatus" : "org:charity" ,
    "org:hasLegalName" : "United Way of Greater Toronto" ,
    "org:numberOfEmployees" : "100" ,
    "org:hasRole" : ["uwgt:ceo", "uwgt:vp_research"]
}
```

4.4. cids:Organization in Turtle

The following depicts the United Way Organization in Turtle[4], a serialization format for RDF. All attribute names correspond to the attributes defined for a cids:Organization.

```
@prefix        caco : <http://ontology.eil.utoronto.ca/CAO/caco#> .
@prefix        cafo : <http://ontology.eil.utoronto.ca/CAO/cafo#> .
@prefix        sch : <http://schema.org/> .
@prefix        org : <http://ontology.eil.utoronto.ca/organization#> .
@prefix        ic : <http://ontology.eil.utoronto.ca/icontact#> .
@prefix        uwgt : <https://www.unitedwaygt.org/LD#> .

uwgt rdf:type cids:Organization ;
        org:hasID  <https://www.unitedwaygt.org/> ;
        ic:hasAddress  uwgt:uwgt_address ;
        org:hasLegalName  "United Way of Greater Toronto" ;
        org:hasLegalStatus. org:charity ;
        org:numberOfEmployees  100 ;
        org:hasRole   [uwgt:ceo, uwgt:vp_research] .
```

[4] https://www.w3.org/TR/turtle/

5. Acknowledgements

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

6. References

CED, (2009), “A guide to Social Return on Investment”, The Canadian CED Network.

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., Barbuceanu, M., Gruninger, M., and Lin, J., (1998), “An Organisation Ontology for Enterprise Modeling”, In *Simulating Organizations: Computational Models of Institutions and Groups*, M. Prietula, K. Carley & L. Gasser (Eds), Menlo Park CA: AAAI/MIT Press, pp. 131-152.

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. (2015) “The Role of Ontologies in Publishing and Analyzing City Indicators”, *Computers, Environment and Urban Systems, Vol. 54, pp. 266-279.*

Horridge, M., Drummond, N., Goodwin, J., Rector, A. L., Stevens, R., & Wang, H. (2006, November). The Manchester OWL syntax. In OWLed (Vol. 216).

IMP, “What is Impact”, Impact Management Project. https://impactmanagementproject.com/impact-management/what-is-impact/

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

---

[i] 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.

[ii] “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/).

[iii] 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 cannot be interpreted by software applications. The latter provides a computationallly precise definition that can be understood by both human and machine.