The Common Impact Data Standard

An Ontology for Representing Impact

Version 2.1, May 2023

Ontology URL: http://ontology.commonapproach.org/owl/cids_v2.1.owl

Namespace: http://ontology.commonapproach.org/

Suggested Prefix: cids

For further information, contact: info@commonapproach.org

Recommended Attribution:
Common Approach to Impact Measurement (2023) "Technical Guide: The Common Impact Data Standard V2.1 An Ontology for Representing Impact", Common Approach to Impact Measurement.
DOI 10.5281/zenodo.7930431
https://www.commonapproach.org/wp-content/uploads/2021/03/Common-Impact-Data-Standard-V2.1.pdf

COMMON APPROACH
TO IMPACT MEASUREMENT

Acknowledgements

The Common Impact Data Standard Version 2.1 was updated by

- Alicia Richins, Common Approach to Impact Measurement, commonapproach.org
- Mark Fox Centre for Social Services Engineering, University of Toronto, csse.utoronto.ca
- Alex Goel, Logical Outcomes
- Dan Lammot, threshold.world and b.world
- Jaclene Roshan, Salesforce Nonprofit Cloud
- Kate Ruff, Common Approach to Impact Measurement, commonapproach.org

With contributions from Meg Gray, Salesforce; Jess Peters and Derek Hatchard, Riddl; Seonaid Lee

Version 2.1 was supported, in part, by the Government of Canada, Investment Readiness Program.

Version 1.1 was principally authored by

- Mark Fox Centre for Social Services Engineering, University of Toronto, csse.utoronto.ca
- Kate Ruff, Common Approach to Impact Measurement, commonapproach.org
- Bart Gajderowicz Centre for Social Services Engineering, University of Toronto, csse.utoronto.ca
- Anshula Chowdhury, Sametrica, sametrica.com
- Tawfiq Abdulai, Sametrica, sametrica.com
- Jane Zhang, Centre for Social Innovation, socialinnovation.org

With contributions from: James Hicks and Spencer Powell, Impact Management Project, and Daniela Rosu, University of Toronto.

Version 1.1 was supported, in part, by the Ontario Ministry of Economic Growth and Development and the Government of Canada, Investment Readiness Program.

Revision History

Revision	Date	Changes
1.1	20 November 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 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:
1.2	20 January 2021	Added object property hasTimeInterval Added "hasTimeInterval only DateTimeInterval" to ImpactReport
		Added "hasImpactReport only ImpactReport" to Outcome
		Changed hasIndicator to forIndicator in ImpactDepth, ImpactDuration and ImpactScale
		In ImpactDuration replace hasStartTime and hasEndTime with "hasTimeInterval exactly 1 time:DateTimeInterval"
		In ImpactNorms, added "hasIndicatorReport only IndicatorReport"
	22 January 2021	Replaced in IndicatorReport i72:for_time_interval with hasTimeInterval (functional)
		Added time:DateTimeDescription class to Section 3.1, with value restriction on hasBeginning and hasEnd being time:DateTimeDescription
		Added org:hasName to ImpactReport
		Replaced schema:description with hasDescription – defined as a functional subproperty of schema:description.
		Replaced schema:name with org:hasName– defined as a functional sub-property of schema:name
		Replaced schema:identifier with org:hasIdentifier– defined as a functional subproperty of schema:identifier

	15 February 2021	Added forStakeholder object property to StakeholderCharacteristic
		Added SocialPurposeOrganization, FinancialOrganization and StandardsOrganization as subclasses of Organization
		Updated prefix list in table 3
		Changed value restriction of hasOccupation in Person to xsd:string
	13 May 2021	Added codeValueDescription to Domain Class in Outcome section
		Updated the JSON-LD example in Section 4 to use the latest context definition files.
		Changed value restriction of forOutcome in ImpactReport to “exactly 1 StakeholderOutcome”.
		Added “time:hasTime exactly 1 time:DateTimeInterval” to ImpactReport
		StakeholderCharacteristic: ● Renamed Characteristic ● Replaced hasStakeholderCharacteristics with hasCharacteristic
	27 June 2021	Added “forOutcome exactly 1 Outcome” to StakeholderOutcome in order to identify which outcome it is associated with
		Updated Common Impact Data Standard graph
		hasImpactReport now appears in both Outcome and StakeholderOutcome
		Added Code a a generic class to refer to standards defined by other organizations, e.g., indicators, outcomes, characteristics, etc.
		Added hasContact to cids:Organization
		BeneficiaryStakeholder renamed to BeneficialStakeholder
		Impact scale, duration, depth generalized into HowMuchImpact.
2.0	August 2021	Added graphs to sections.
	14 November 2021	All prov:wasGeneratedBy value restrictions changed to Activity
2.1	23 February 2022	Minor miscellaneous corrections.
	1 March 2022	Change Stakeholder so that it is no longer a subclass of Person or Organization.
	1 April 2022	Add hasDescription to Characteristic.
	27 April 2022	Removed hasCharacteristic from LogicModel
	18 August 2022	Added reference to linked data in 1.3 Added “hasLocation only Feature” to cids:Organization Added JSON-LD explanation
	20 January 2023	“Domain” renamed “Theme” Added internally derived themes to the definition for Outcome Themes In IndicatorReport, opened up prov:Generated by value restriction to capture more than one method
	April 2023	Minor miscellaneous corrections Added gn:Feature so that Stakeholder can be a organization, person or feature, such as the planet, a river, or a forest. Updated all Figures

COMMON APPROACH
TO IMPACT MEASUREMENT

Table of Contents

How to read this document

The Common Impact Data Standard specifies an ontology for impact measurement. The ontology is currently implemented in Web Ontology Language (OWL). Developers may find the OWL file most useful. Others will find this document a more readable format. Both this document and the OWL file include all the information you need to understand and implement the standard.

This document, which is written and maintained by the technical committee, has three main purposes:

1. It communicates specifications for the Common Impact Data Standard.
2. It contains a detailed description of the standard for software vendors who are interested in providing impact measurement applications for their clients.
3. It gives software developers a technical introduction to the data standard and orients them in the field of impact measurement.

The goal is to ensure that those in various roles within software development organizations have a common understanding of the problem the data standard aims to address and the nature of the solution.

Those in different roles may need to take a different approach to reading the document.

Everyone should read and understand Section 1: Introduction to the Common Impact Data Standard, which establishes the motivation for the standard.

If you are in a management role at a software company, and unlikely to write the code yourself, you may benefit from reading Section 2: Common Impact Data Standard; however, we recommend you skip these formal tables and focus on Class descriptions and diagrams. You will particularly want to focus on Figure 3 (at the beginning of Section 2), which shows relationships at a high level and how they demonstrate an organization's impact.

If you are a developer, you will need more detail, and you should acquaint yourself with the individual Class tables in Section 2: Common Impact Data Standard. You should also make sure you understand the conceptual mappings laid out in Section 1.3, as they will allow you to better understand your users' requirements. Section 3: Foundation Ontologies provides documentation for external ontologies that describe categories of knowledge, such as time measurement and addresses. If you track this level of detail in your implementation, these classes and properties are available for defining precise relationships that underlie or elaborate upon your Impact Measurement information.

1. Introduction to the Common Impact Data Standard

The Common Impact Data Standard (Data Standard) is a standardized way to represent a social purpose organization's impact model (i.e. a theory of change, logic model, outcome chain, etc.) and the effects of their work on people and the planet.

The purpose of the Common Impact Data Standard is to create a shared data model with which to capture, export, and import data about impact measurement. The standard is essentially a set of detailed descriptions of how data should be structured to allow interoperability with other aligned software products. This approach allows data to be moved between applications without any additional mapping.

The Data Standard was developed in consultation with experts and other leading standards, such as the Impact Management Project (now housed at Impact Frontiers). It maps to prevailing impact models. Software that includes the Common Impact Data Standard can be used to capture the description of a social purpose organization's work using the language and models they use to describe themselves.

This is important for situating your work in terms that your potential clients already understand.

Ultimately, the Common Impact Data Standard will facilitate flexible, shareable impact data that allows each social purpose organization to design an impact model that is most relevant to it and its stakeholders.

1.2. Benefits of a Common Impact Data Standard

The benefits of a Common Impact Data Standard are:

1. **Sophisticated analysis**: The Data Standard makes it possible for researchers to integrate their data, thereby enabling a plethora of analysis, e.g., longitudinal and transversal studies, using a variety of methods. This may lead to a better understanding of needs and a better understanding of what works.

2. **More autonomy**: Donors, investors, and government agencies are increasingly aware that old impact reporting techniques have been a burden to grantees and investors. A common impact data standard provides funders the standard definitions that they need to understand portfolio-level impacts, while leaving social purpose organizations the autonomy to measure impact in ways that best-fit the social purpose organizations own data needs.

3. **Less paperwork**: A common impact data standard allows impact data to be represented in ways that can accommodate the reporting needs of diverse funders.social purpose organizations utilizing a common impact data standard will need to do less custom reporting.

4. **Greater visibility:** A common impact data standard enables the tagging of an organization's impact content on the internet making it easier for search engine users to find impact content on the web.

5. **More versatility:** A common impact data standard makes it easier for organizations to benchmark their impact measurement with other measurement standards, and understand their impact in the context of those other standards, such as the UN SDG Global Indicator Framework, IRIS+ and the International Aid Transparency Initiative (IATI) Standard.

6. **Better impact:** Each organization makes some difference, but more opportunities to create greater change can be created when the data can be connected and aggregated. A common impact data standard allows networks to pool data, to see impact and use the data to improve impact.

1.3. What is the "impact" that the Common Impact Data Standard represents?

We understand **impact** as a change in outcomes to people and the planet.

The Common Impact Data Standard builds on the Impact Management Norms model, which names five dimensions of Impact: **what**, **who**, **how much**, **contribution**, and **risk**. To this, the Common Impact Data Standard adds a sixth dimension of **how**. "How" represents the efforts that a social purpose organization undertakes to achieve certain Outcomes.

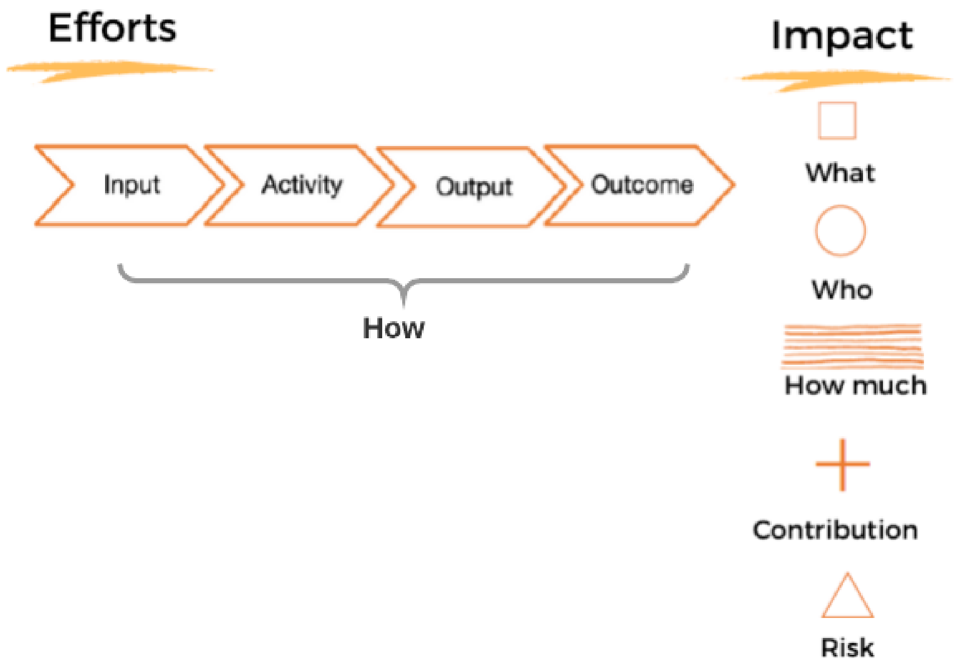


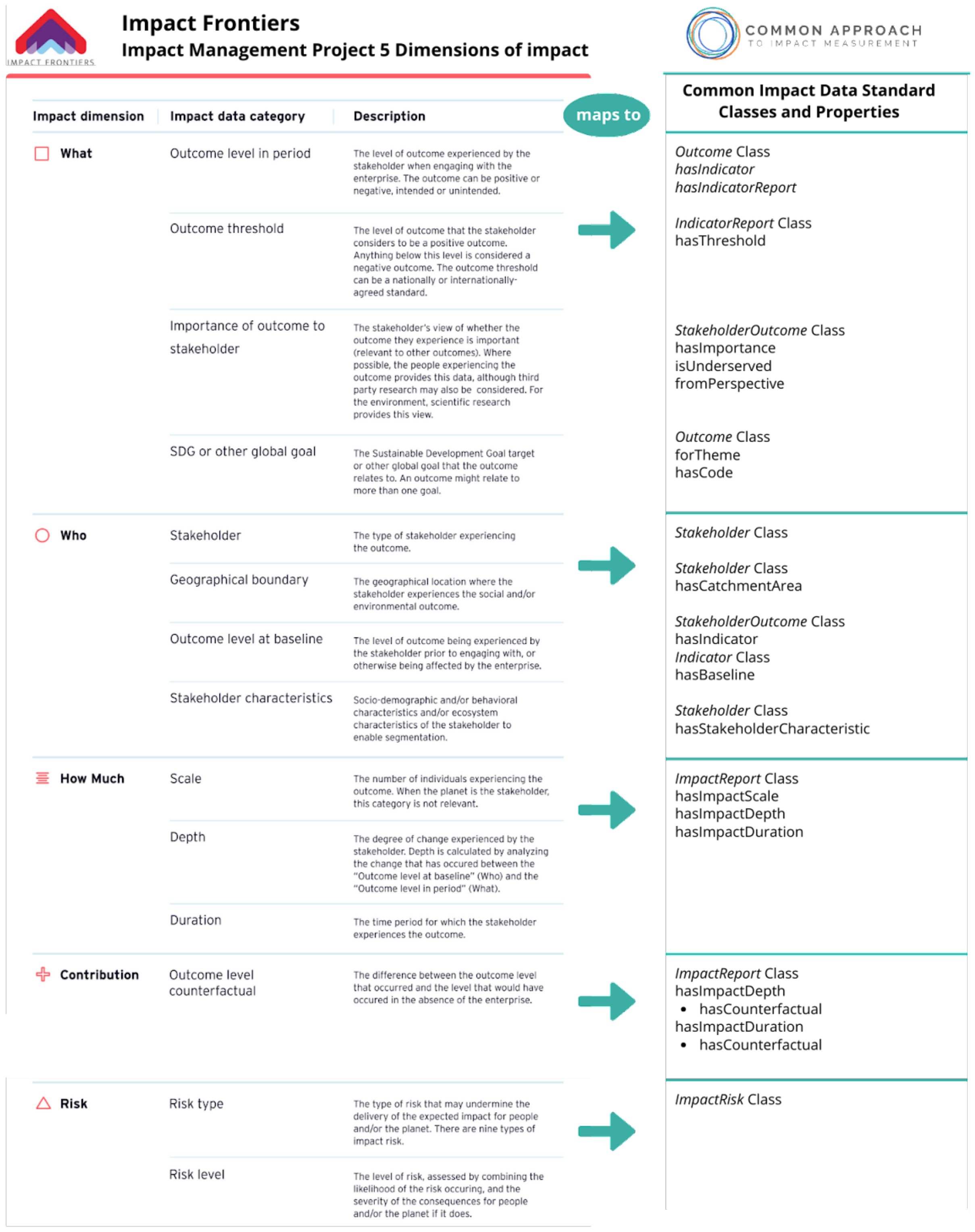
Figure 1: Summary of the impact representation specified in the Common Impact Data Standard. The Common Impact Data Standard represents the five dimensions of impact as defined by the Impact Management Project – what, who, how much, contribution and risk - plus a sixth dimension - how.

The classes and properties in the Common Impact Data Standard align with prominent impact standards and methods. This makes it possible for social purpose organizations to share data amongst themselves, regardless of the variation they use.

To represent the processes by which a social purpose organization delivers *Outcomes* to its *Stakeholders*, the Common Impact Data Standard uses the following classes:

- *Program* Class
- *Service* Class
- *Activity* Class
- *Input* Class
- *Output* Class
- *Outcome* Class
- *Indicator* Class
- *IndicatorReport* Class

To represent the what, who, how much, contribution and risk dimensions of impact, Common Approach uses the Classes shown in Figure 2.



Impact Frontiers
Impact Management Project 5 Dimensions of impact

COMMON APPROACH TO IMPACT MEASUREMENT

maps to

Impact dimension	Impact data category	Description	Common Impact Data Standard Classes and Properties
☐ **What**	Outcome level in period	The level of outcome experienced by the stakeholder when engaging with the enterprise. The outcome can be positive or negative, intended or unintended.	*Outcome* Class *hasIndicator* *hasIndicatorReport*
	Outcome threshold	The level of outcome that the stakeholder considers to be a positive outcome. Anything below this level is considered a negative outcome. The outcome threshold can be a nationally or internationally-agreed standard.	*IndicatorReport* Class hasThreshold
	Importance of outcome to stakeholder	The stakeholder's view of whether the outcome they experience is important (relevant to other outcomes). Where possible, the people experiencing the outcome provides this data, although third party research may also be considered. For the environment, scientific research provides this view.	*StakeholderOutcome* Class hasImportance isUnderserved fromPerspective
	SDG or other global goal	The Sustainable Development Goal target or other global goal that the outcome relates to. An outcome might relate to more than one goal.	*Outcome* Class forTheme hasCode
○ **Who**	Stakeholder	The type of stakeholder experiencing the outcome.	*Stakeholder* Class
	Geographical boundary	The geographical location where the stakeholder experiences the social and/or environmental outcome.	*Stakeholder* Class hasCatchmentArea
	Outcome level at baseline	The level of outcome being experienced by the stakeholder prior to engaging with, or otherwise being affected by the enterprise.	*StakeholderOutcome* Class hasIndicator *Indicator* Class hasBaseline
	Stakeholder characteristics	Socio-demographic and/or behavioral characteristics and/or ecosystem characteristics of the stakeholder to enable segmentation.	*Stakeholder* Class hasStakeholderCharacteristic
☰ **How Much**	Scale	The number of individuals experiencing the outcome. When the planet is the stakeholder, this category is not relevant.	*ImpactReport* Class hasImpactScale hasImpactDepth hasImpactDuration
	Depth	The degree of change experienced by the stakeholder. Depth is calculated by analyzing the change that has occured between the "Outcome level at baseline" (Who) and the "Outcome level in period" (What).	
	Duration	The time period for which the stakeholder experiences the outcome.	
✚ **Contribution**	Outcome level counterfactual	The difference between the outcome level that occurred and the level that would have occured in the absence of the enterprise.	*ImpactReport* Class hasImpactDepth • hasCounterfactual hasImpactDuration • hasCounterfactual
△ **Risk**	Risk type	The type of risk that may undermine the delivery of the expected impact for people and/or the planet. There are nine types of impact risk.	*ImpactRisk* Class
	Risk level	The level of risk, assessed by combining the likelihood of the risk occuring, and the severity of the consequences for people and/or the planet if it does.	

Source: Impact Management Project , housed at Impact Frontiers

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

1.4. Language and ontologies of the Common Impact Data Standard

1.4.1. OWL

The Common Impact Data Standard is authored in the Web Ontology Language (OWL), which is used to formally define taxonomy and classification networks for publishing on the Semantic Web. That is to say, OWL provides a machine-readable way to describe:

- What things exist and
- How they relate to one another
- In a particular area of practice.

By definition, an OWL ontology conforms to Linked Data requirements (Poblet et al., 2019).

The OWL specifications are laid out in Ontology Tables in Section 2 and Section 3.

1.4.2. Common Impact Data Standard and Foundational Ontologies

The Common Impact Data Standard is fully described in Section 2.

The ontology has been developed by Common Approach to facilitate describing impact measurement in the terms used by researchers and social purpose organizations. It is a high-level conceptual frame that reports at the level of the Impact Management Norms.

Some software vendors may already provide or wish to provide a more fine-grained analysis of programs and activities. The Foundation Ontology documentation (provided in Section 3) describes a set of existing standards which have been included to support such features. They are included in this document for completeness and convenience, but the requirements of Impact Measurement are captured by the Classes in Section 2.

Notes for developers

Each of the ontologies used in this project is represented with a short form that can be used as a prefix to indicate where the foundational ontology is published. For example, "ic" refers to the International Contacts Ontology, which provides a standard for storing addresses and phone numbers (available at http://ontology.eil.utoronto.ca/icontact.owl). The prefixes used are listed in Appendix 2.

The prefixes are part of the name of the Class. Where no prefix is included, it can be assumed to be a part of the Common Impact Data Standard, although "cids" may also be used if a prefix is included for disambiguation. This means that **Activity and act:Activity refer to different Class descriptions and are not interchangeable**. However, cids:Activity and Activity *are* two ways of representing the same Class.

Appendix 2 lists the prefixes used in this document.

As in object-oriented languages such as Java, the properties of Classes have descent. That is, any Class which is a subClass of another can use any property:value pairs described in the superClass.

For example, LogicModel has access to all the properties contained in ImpactModel, as well as the ones explicitly added in its own class definition. This is also true for Core ontology members that have a superClass from the Foundation ontology (such as the previously-mentioned Activity, which is a subClass of act:Activity.)

1.4.3. Classes

Each Class describes a type of thing (e.g. Organization, Activity, Indicator) and includes a list of rules about how to describe it, including where it fits in the conceptual framework.

In this document, each Class is described by a table. The tables show formal statements of some combination of:

- How to store values directly (for example, a string that is used as a name)
- Relationships between objects (that is, links to other Classes)
- Constraints on what is allowed (for example, on how many of each type of thing can or must exist)

Even if you are new to OWL, the tables should still be relatively self-explanatory for those with a technical background. The constraints on the values are similar to those you would find in a relational database, and the descriptions of classes look similar to objects in OOP.

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. They can be found in the OWL file at http://ontology.commonapproach.org/owl/cids_v2.1.owl.

The following table is an example. The class column specifies the name of the class being defined. This table describes the Organization class—this is a partial list for explanatory purposes and should not be used as a reference. The complete description of the Organization class is in Section 2.5

The first line states that Organization is a subclass of org:Organization, which is defined in the foundational ontologies (Section 3). 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	Value Restriction
Organization	rdfs:subClassOf	org:Organization
	hasName	only xsd:string
	sch:legalName	exactly 1 xsd:string
	ic:address	max 1 ic:Address

Table 1: Partial representation of the class Organization (example only).

The following value restrictions are used in this document:

- "min n": Specifies that the property has to have a minimum of n values.
- "max n": Specifies that the property has to have a maximum of 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.

Any instance of a class must satisfy the class's definition, including conforming to any restrictions on properties and values. 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 sch:legalName 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	cids:Organization
	hasName	"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

1.4.4. Identifying Relationships between Things

You will see that the "Value" in a Property may be an instance of another Class. These relationships form the links in the graph and can be used to describe rich and flexible networks of activities, programs, and impacts across the whole range of an organization's stakeholders.

Some of the most important connections are illustrated in the accompanying figures—these are not comprehensive but are offered to ease the process of reading the ontology. When examining the figures, you should focus on the highlighted Classes, which are explained in that section. Other relationships are expanded upon in other sections.

You should note that property/value restrictions describe relationships between instances in a graph, not between the Classes themselves.

2. Common Impact Data Standard

In this section, we define the classes that comprise the Common Impact Data Standard Core Ontology. Figure 3 depicts the main classes and a subset of the properties that connect them. The classes and properties together are sufficient for representing any sort of impact model, be it a logic model, theory of change, outcomes map or impact map. Classes in yellow define the impact model, classes in white report on the performance of the organization's activities with respect to their impact model. For simplicity, this depiction does not portray all of the relationships between classes.

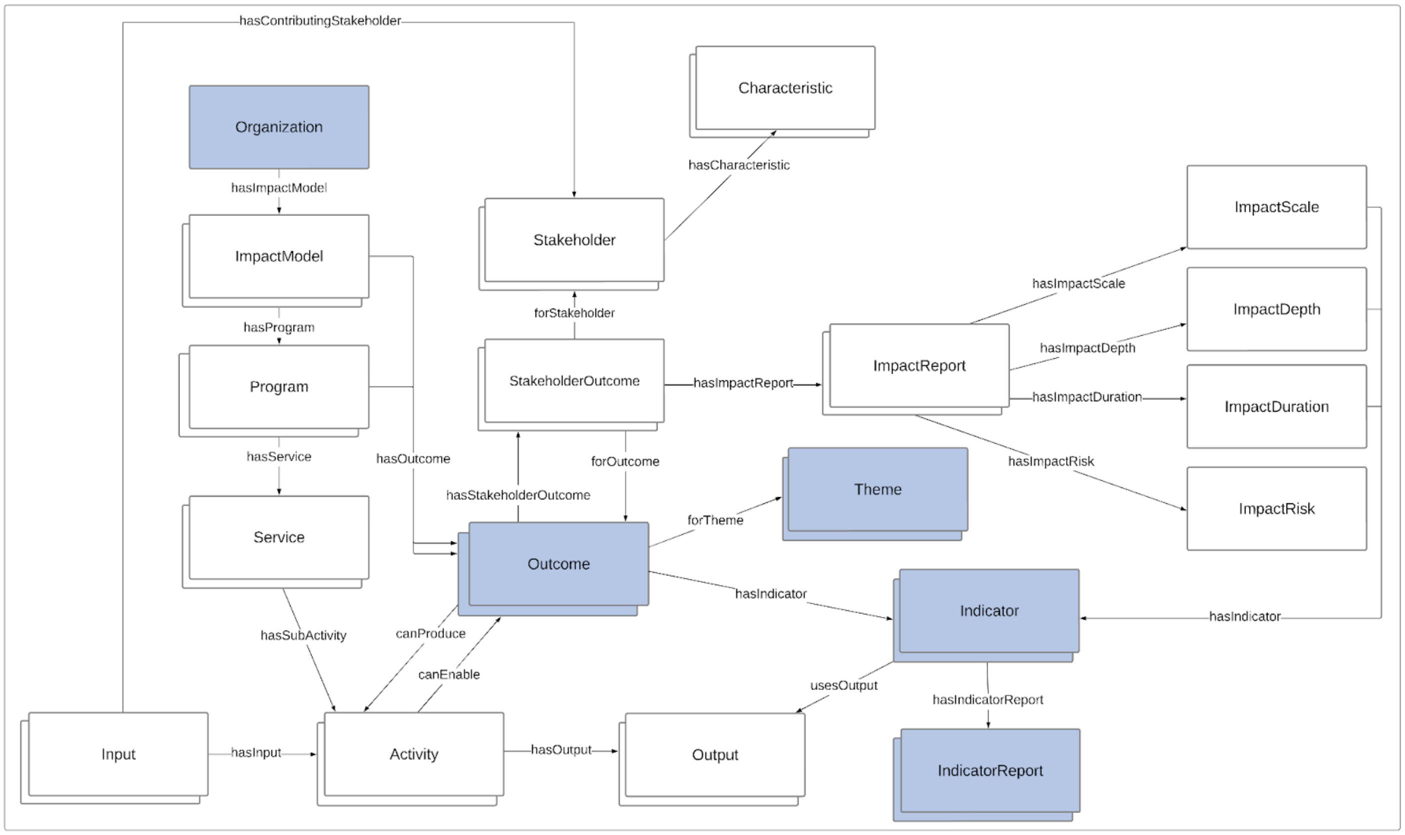


Figure 3: Detail of the impact representation specified in the Common Impact Data Standard. The Common Impact Data Standard represents five dimensions of Impact as well as the components of impact models, including activity, output, and outcome. The shaded classes are those required for Basic Tier alignment.

2.3. ImpactModel

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, Results Framework, and Outcomes Map.

The Common Impact Data Standard is designed to represent the different variations of impact models in use by Social Purpose Organizations. Of course, many software vendors only implement one model.

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.

The following graph (Figure 4) depicts the main classes and properties of an Impact Model.

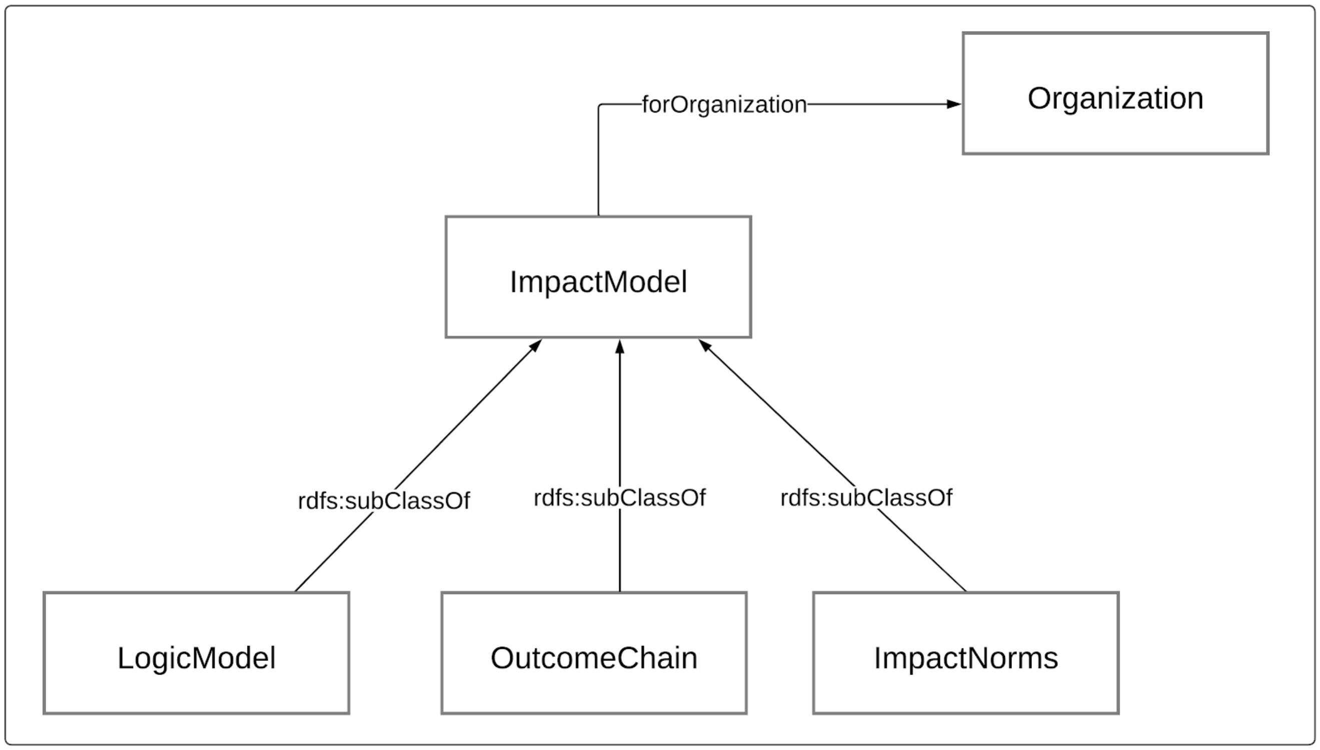


Figure 4: Impact Model Pattern. This Pattern shows that the Common Impact Data Standard supports the representation of many different impact models

Note that in the descriptions that follow, we use the word *key*. By *key* we mean a subset of each that is important to depict at this level of abstraction, i.e. critical, material, fundamental, etc. 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:

- hasName: A string containing a title for the model.
- hasDescription: 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	hasName	exactly 1 xsd:string
	hasDescription	exactly 1 xsd:string
	sch:dateCreated	exactly 1 xsd:date
	forOrganization	exactly 1 Organization

2.3.1. LogicModel

LogicModel is a subclass of ImpactModel. Its properties link to the main impact modelling classes used by a Logic Model. 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

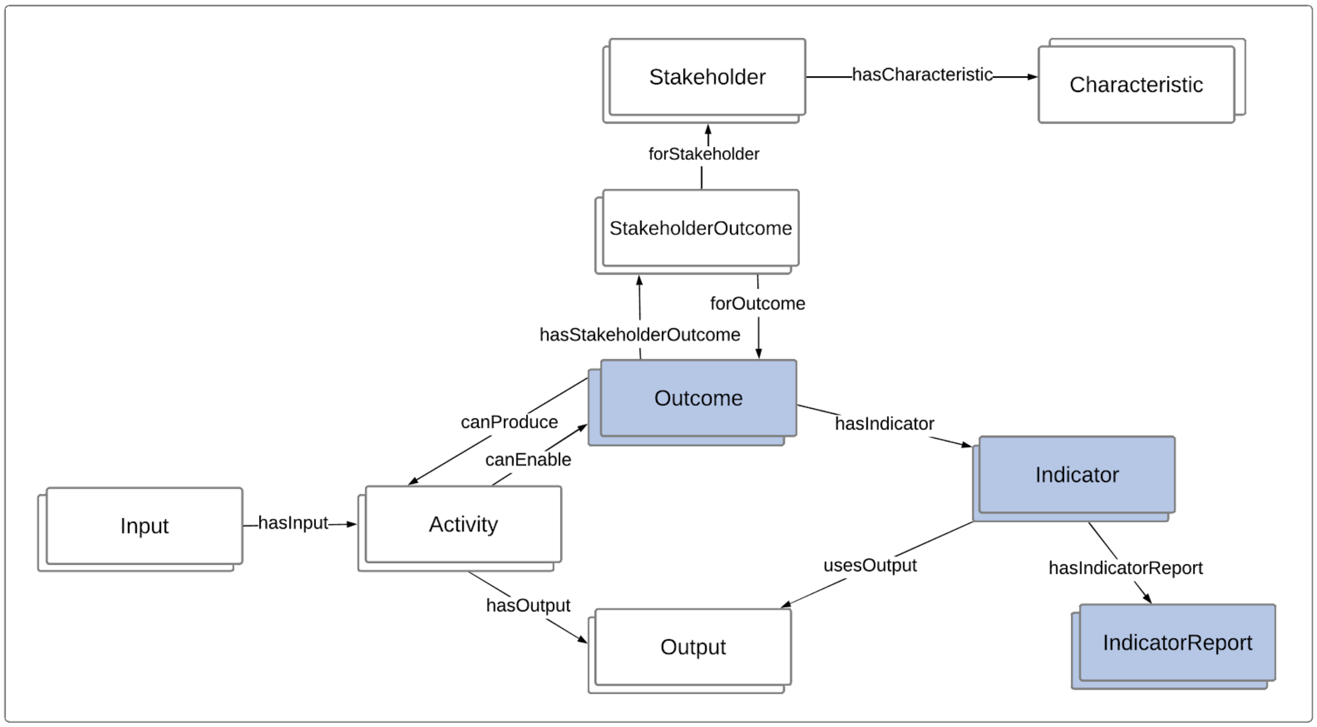


Figure 5: Logic Model Pattern. This Pattern shows how the classic Logic Model or Theory of change can be represented using the Common Impact Data Standard. Activities have inputs. Activities have Outputs and lead to Outcomes. Outcomes have indicators which are reported using an IndicatorReport. Outputs are metrics which may be used to express Outcome indicators. Note that many illustrations of the LogicModel present Outputs as leading to Outcomes. Common Impact Data Standard defines Outputs as a measure of the amount of Activity, and models Activities leading to Outcomes.

2.3.2. OutcomeChain

OutcomeChain is a subclass of ImpactModel. Its properties link to instances of the main impact modelling classes it uses. 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

2.3.3. ImpactNorms

ImpactNorms is a subclass of ImpactModel and represents the Impact Management Norms defined by the Impact Management Project, now hosted by Impact Frontiers.

The top-level definition of an ImpactNorms contains the following properties:

- hasStakeholder: Identifies stakeholders which are the focus of the outcomes – corresponds to the "Who" in the Impact Measurement Norms Five Dimensions of Impact.
- hasOutcome: Identifies key Outcomes for the model – corresponds to the "What" in the Impact Measurement Norms Five Dimensions of Impact.
- hasStakeholderOutcome: Identifies the outcome and the importance of the outcome to the stakeholder. This defines a **who-what** pairing. Impact Measurement Norms Five Dimensions of Impact measures **how much, contribution** and **risk** for each **who-what** pair.
- hasImpactReport: Identifies key Impacts for the model. This includes the **how much**, **contribution** and **risk** for each StakeholderOutcome.
- hasIndicator: Identifies key Indicators for the model.
- hasIndicatorReport: A set of IndicatorReports associated with this model.

Class	Property	Value Restriction
ImpactNorms	rdfs:subClassOf	ImpactModel
	hasStakeholder	only Stakeholder
	hasOutcome	only Outcome
	hasStakeholderOutcome	only StakeholderOutcome
	hasIndicator	only Indicator
	hasImpactReport	only ImpactReport
	hasIndicatorReport	only IndicatorReport

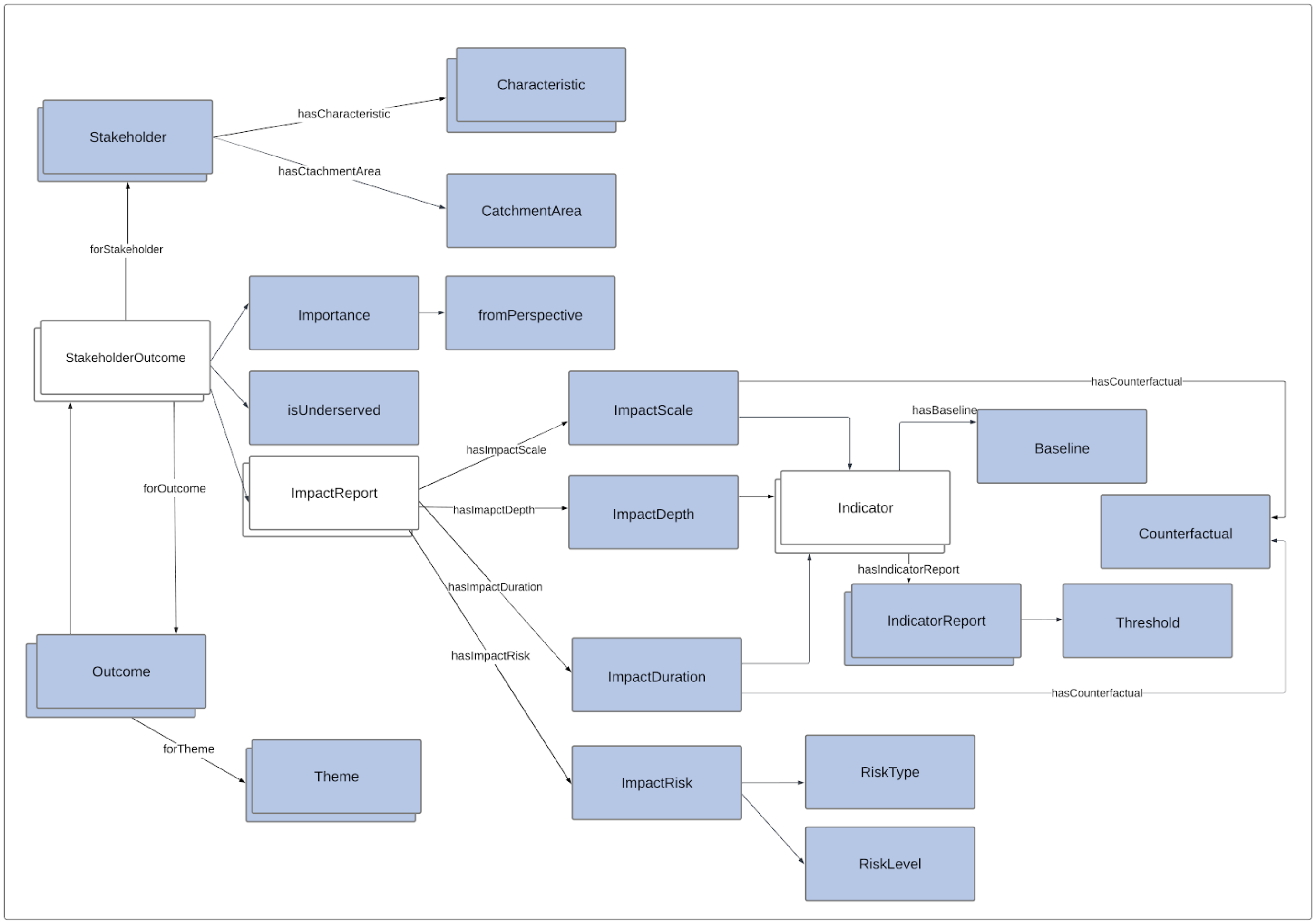


Figure 6: ImpactNorms Pattern. This Pattern shows how the Impact Management Norms, specifically the five dimensions of Impact and associated data categories, can be represented using the Common Impact Data Standard. Shaded boxes represent the data categories as mapped in Figure 2.

2.4. Code

The Common Impact Data Standard is designed to work with other standards. There are many schema and taxonomies to support impact measurement. To support interoperability between standards, the Common Impact Data standards uses the Code class. The Code class is used to reference taxonomies of outcomes, stakeholder characteristics, themes, indicators, etc. Taxonomies can be externally defined, as in the SDGs or IRIS+ Indicators; or they can be internally defined, such as a list of StakeholderCharacteristics. An instance of Code is used to represent a specific code in a codelist by specifying the source codelist, code definition and code identifier. It has the following properties:

- definedBy: The Organization that defined the code.
- hasSpecification: The URI where the definition of the code can be found, including its version.
- hasIdentifier: The code's unique identifier.
- hasName: A name or title for the code.
- hasDescription: A description of the code.
- schema:codeValue: The value of the code, if appropriate, specified as a string.
- i72:value: Alternative specification of codeValue using the ISO/IEC 21972 Measure.

hasIdentifier, hasName, hasDescription and codeValue act as a cache of the equivalent information to be found in the specification of the Code referred to by hasSpecification.

The following table defines the Code class.

Class	Property	Value Restriction
Code	definedBy	exactly 1 Organization
	hasSpecification	exactly 1 xsd:anyURI
	org:hasIdentifier	exactly 1 xsd:string
	hasName	exactly 1 xsd:string
	hasDescription	exactly 1 xsd:string
	schema:codeValue	exactly 1 xsd:string
	i72:value	exactly 1 i72:Measure

2.5. Organization

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:

- org:hasID: Identifies one or more org:OrganizationID.
- hasImpactModel: Identifies one or more ImpactModels for the organization.
- hasIndicator: List of indicators associated with the Organization.
- hasOutcome: List of outcomes associated with the Organization.
- hasCharacteristic: List of stakeholder characteristics associated with the Organization.
- hasContact: Identifies one or more contacts for the organization.
- hasDescription: a string that describes the Organization.
- hasWebAddress: a URI specifying the website for the organization.
- hasMission: text defining the purpose of the organization.
- sch:dateCreated: A xsd:date the organization was created, in the format of yyyy-mm-dd.

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
	org:hasID	only org:Organization ID
	hasImpactModel	only ImpactModel
	hasIndicator	only Indicator
	hasOutcome	only Outcome
	hasCharacteristic	only Characteristic
	hasContact	only Person
	hasDescription	exactly 1 xsd:string
	hasWebAddress	only xsd:anyURI
	hasMission	exactly 1 xsd:string
	sch:dateCreated	exactly 1 xsd:dateTime
SocialPurposeOrganization	rdfs:subClassOf	Organization
FinancialOrganization	rdfs:subClassOf	Organization
StandardsOrganization	rdfs:subClassOf	Organization

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

- org:hasID: Ties to a unique recognized/public identifier for the Organization i.e. a business number, charity number, etc.
- ic:has Address: The main address of the Organization.
- org:hasLegalName: Is 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.

Note that org:FormalOrganization is a subclass of org:Organization to distinguish it from informal organizations. No additional properties are defined but can be extended by the user.

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:FormalOrganization	rdfs:subClassOf	org:Organization
org:OrganizationID	org:issuedBy	org:Organization
	hasIdentifier	exactly 1 xsd:string
	sch:dateCreated	exactly 1 xsd:dateTime

2.6. Outcome, StakeholderOutcome, and ImpactReport

An outcome is the level of well-being experienced by a group of people, or the condition of the natural environment, as a result of an event or action (Source: Impact Management Norms). Outcomes are what stakeholders experience as a result of an organization's activities. They can be positive or negative, intended or unintended.

Impact Management Norms' Five Dimensions of Impact and Data Categories require a What-Who combination. For example, the outcome level in the period refers to the level experienced by a particular stakeholder group. It is not just the level of the outcome (what) but the level experienced by the stakeholder (who). The measured level refers to a specific what-who combination. Similarly, the importance of an outcome to a stakeholder is not just the importance of the outcome (what) but to a specific stakeholder (who). The class StakeholderOutcome represents this What-Who combination.

- An organization may define several outcomes and several stakeholders. The organization may wish to report how different stakeholder groups (e.g. children, men, women, newcomers) experience the outcome (e.g. food security). StakeholderOutcome creates the who-what combination to represent the level of outcome experienced by a specific stakeholder group.
- To help ensure that organizations are listening to and meeting the needs of stakeholders, Impact Management Norms recommend that organizations know and disclose the importance of the outcome to each stakeholder (e.g., "high importance," "moderate important," "neutral," or "unimportant"), and from whose perspective its importance was assessed (fromPerspectiveOf). This is represented by StakeholderOutcome hasImporance. When the stakeholder is unable to express importance, the importance may be from the perspective of another entity. For example, the importance of an outcome to the planet may be from the perspective of the Intergovernmental Panel on Climate Change (IPCC). Similarly, the importance to a child might be assessed from the perspective of the parents. FromPerspectiveOf can also be used to indicate when the social purpose organization has estimated the importance of the outcome on behalf of the stakeholder (though this is not ideal).
- Targets, baselines and contributions can be represented for each StakeholderOutcome. Since StakeholderOutcome refers to a specific Stakeholder, a more specific outcome and set of indicators can be defined and used to measure the impact (hasIndicator), and the report of actual impact (hasImpactReport).

2.6.1. Outcome

Figure 7 depicts the main classes and properties of the Outcome pattern.

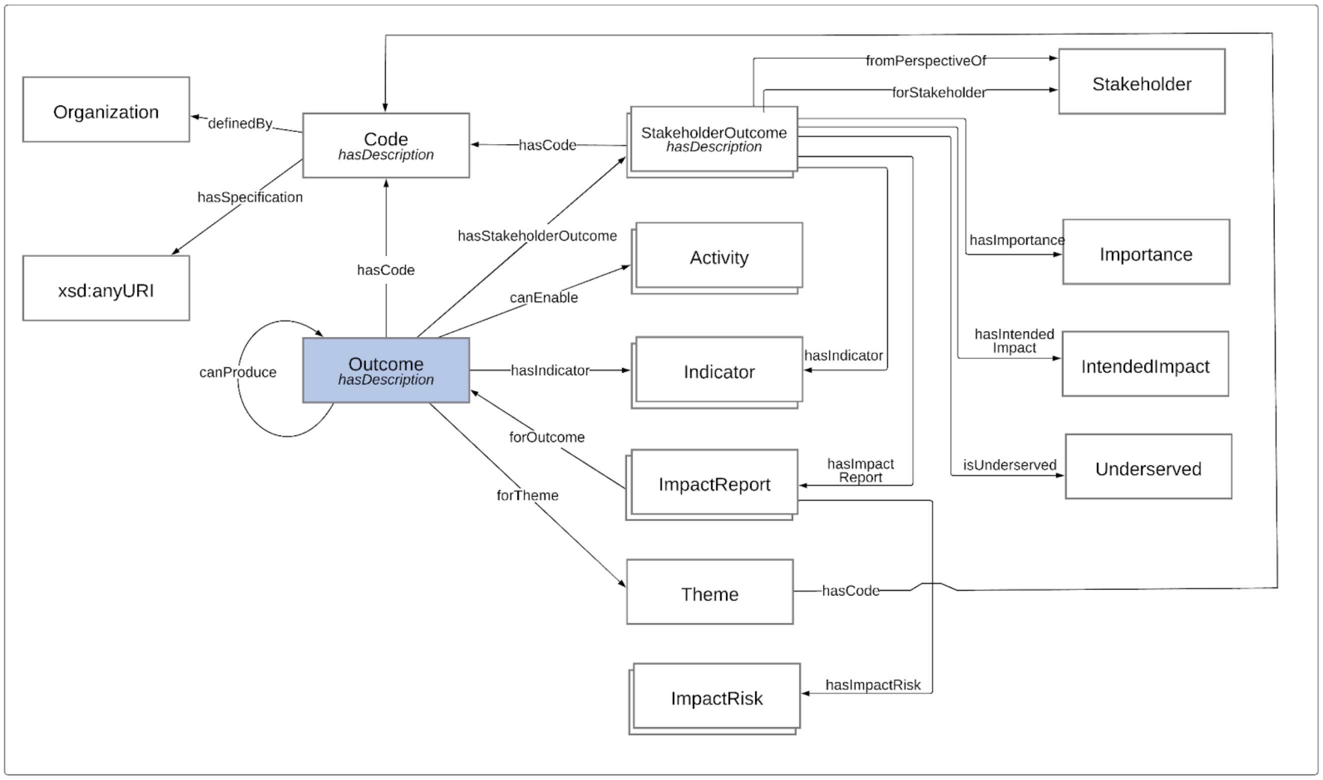


Figure 7: Outcome Pattern

An Outcome has the following properties:

- hasCode: Links to zero or more Codes that are externally defined taxonomies of outcomes.
- hasStakeholderOutcome: Identifies the impact it has on Stakeholders.
- forTheme: Identifies the externally or internally defined themes that the Outcome aligns with (eg. UNSDG2 or "Food security"),
- 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.
- canProduce: Links an Outcome to another Outcome. It abstracts the underlying activity chain that usually links one Outcome to another.
- i72:located_in: Identifies the spatial location the outcome is defined for.
- oep:partOf: Identifies the Impact Model it is a component of.
- hasName: Identifies the name of the Outcome.

- hasDescription: A description of the Outcome.
- sch:dateCreated: Is the date that the Outcome was created.

Class	Property	Value Restriction
Outcome	hasCode	only Code
	hasStakeholderOutcome	only StakeholderOutcome
	forTheme	only Theme
	hasIndicator	only Indicator
	canEnable	only (Service or Activity)
	canProduce	only Outcome
	i72:located_in	only i72:Feature
	oep:partOf	exactly 1 ImpactModel
	hasName	exactly 1 xsd:string
	hasDescription	exactly 1 xsd:string
	sch:dateCreated	exactly 1 xsd:dateTime

2.6.2. *Theme*

The Theme class is used to represent the impact themes for which outcomes are specified. It has the following properties:

- hasCode: Links to zero or more Codes that are defined taxonomies of themes.
- hasName: A string that provides a name for the theme.
- hasDescription: A string that provides a description for the theme.

The following table defines the Theme class.

Class	Property	Value Restriction
Theme	hasCode	only Code
	hasName	only xsd:string
	hasDescription	only xsd:string

2.6.3. *StakeholderOutcome*

The StakeholderOutcome specifies the outcome for a specific stakeholder, as well as properties of that outcome, such as if the specific stakeholder is underserved with respect to this outcome.

- hasCode: Links to zero or more Codes that are defined taxonomies of stakeholder outcomes.
- forStakeholder: Identifies the Stakeholder affected.
- forOutcome: Identifies the more general outcomes this is part of.
- 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 is underserved in relation to your specific outcome.
- intendedImpact: Identifies the intended direction of the change. Note that ImpactReport captures the actual direction. This helps to inform the interpretation of the ImpactReport.
- 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.
- hasName: A string that is the name of the StakeholderOutcome.
- hasDescription: A string that is description of the StakeholderOutcome.

Class	Property	Value Restriction
StakeholderOutcome	hasCode	only Code
	forStakeholder	exactly 1 Stakeholder
	forOutcome	exactly 1 Outcome
	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
	hasName	only xsd:string
	hasDescription	only xsd:string

2.6.4. ImpactReport

The ImpactReport represents **how much**, **contribution** and **risk** dimensions for each StakeholderOutcome:

- How much of the outcome is occurring – across scale, depth and duration?
- Would this change likely have happened anyway?

How much measures the degree of impact a social purpose organization has on its stakeholders. The ImpactReport allows indicators in the impact report to be classified according to the **how much** dimension of impact norms:

1. **ImpactScale**: refers to indicators that measure the number of individuals who are affected by the SPO's outcome.
2. **ImpactDepth**: refers to indicators that measure the degree of change experienced by the stakeholders compared to some baseline determined prior to the service being provided.
3. **ImpactDuration**: refers to indicators that measure how long the stakeholder experiences, or is likely to experience, the outcome.

Contribution compares the degree of impact against a control group or a baseline. A social purpose organization that is measuring a counterfactual would use the Counterfactual class to specify the spatial area over which the counterfactual was measured, the time interval, method of measurement and the value of the measurement. All indicators in the ImpactReport can be replicated as counterfactual. Scale, depth and duration indicators can each be replicated for the counterfactual.

The ImpactScale, ImpactDepth and ImpactDuration each includes the properties:

- **hasIndicator**: links to the Indicator used to measure the impact
- **prov:wasGeneratedBy**: links to the activity that generated the impact value if an indicator is not specified
- **hasCounterfactual**: links to a Counterfactual which can be used to calculate what the impact on stakeholders would be if the stakeholders did not receive the service.

The Counterfactual class specifies the spatial area over which the counterfactual was measured, the time interval, the method of measurement and the value of the measurement.

Figure 8 depicts the Impact Report Pattern. There are three core classes: ImpactReport which records the impact the service has on stakeholders, Indicator which is used to measure the impact, and Counterfactual which is used to measure stakeholder impact in the absence of the service being provided.

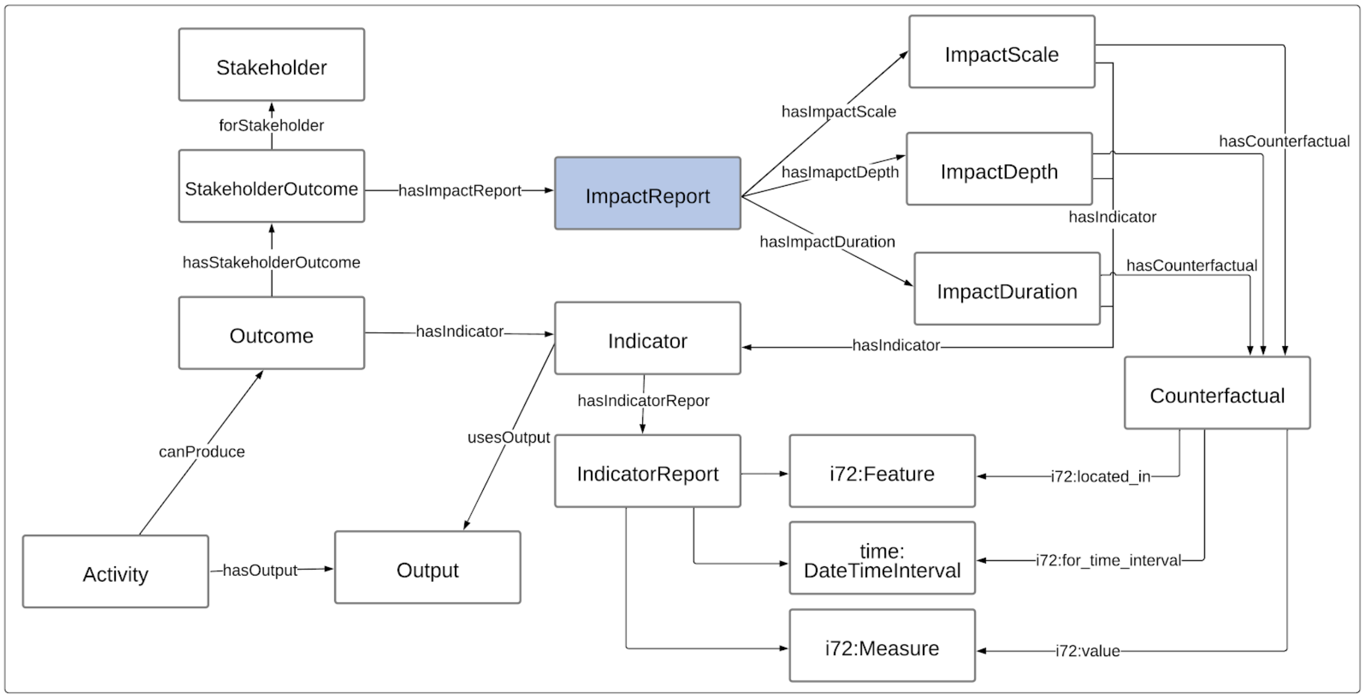


Figure 8: ImpactReport Pattern

An ImpactReport has the following properties:

- forOrganization: Links to the Organization for whom this report is associated.
- time:hasTime: Specifies the time interval the report covers.
- forOutcome: Links to the Organization's Outcome for whom this report is associated.
- hasName: Name or title of the report.
- 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: The degree of difference between the assumed condition that would take place without intervention and the condition with interventions implemented on the stakeholders.
- 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.
- hasImpactRisk: Links to information about Impact Risk as defined by Impact Management Norms.
- hasComment: A string field where any comments that the organization has can be provided.

Class	Property	Value Restriction
ImpactReport	forOrganization	exactly 1 Organization
	time:hasTime	exactly 1 time:DateTimeInterval
	forOutcome	exactly 1 StakeholderOutcome
	hasName	exactly 1 xsd:string
	hasReportedImpact	exactly 1 {“positive”, “negative”, “neutral”}
	hasImpactScale	exactly 1 ImpactScale
	hasImpactDepth	exactly 1 ImpactDepth
	hasImpactDuration	exactly 1 ImpactDuration
	hasImpactRisk	only ImpactRisk
	hasExpectation	exactly 1 xsd:string
	hasComment	exactly 1 xsd:string

2.6.5. *HowMuchImpact*

HowMuchImpact defines the properties common to ImpactScale, ImpactDepth and ImpactDuration:

- forIndicator: Indicator used to measure the impact (not a number value).
- prov:wasGeneratedBy: An activity that can be used to describe the method used to determine the impact. For example, could be a random control trial, or estimation.
- hasDescription: A string that can be 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
HowMuchImpact	forIndicator	exactly 1 Indicator
	prov:wasGeneratedBy	exactly 1 Activity
	hasDescription	exactly 1 xsd:string
	i72:value	exactly 1 i72:Measure
	hasCounterfactual	only Counterfactual
ImpactScale	rdfs:subClassOf	HowMuchImpact
ImpactDepth	rdfs:subClassOf	HowMuchImpact
ImpactDuration	rdfs:subClassOf	HowMuchImpact
	hasTime	exactly 1 time:DateTimeInterval

2.6.6. *Counterfactual*

Counterfactual defines what the impact on stakeholders would be if the stakeholders did not receive the service. It has the following properties:

- i72:located_in: Identifies the spatial location the outcome is defined for.
- time:hasTime: Identifies the time interval during which the counterfactual was determined.
- prov:wasGeneratedBy: An activity that describes how the counterfactual was generated.
- hasDescription: A string that describes the counterfactual.
- i72:value: A measure that specifies the value of the counterfactual.

Class	Property	Value Restriction
Counterfactual	i72:located_in	only i72:Feature
	time:hasTime	exactly 1 time:DateTimeInterval
	prov:wasGeneratedBy	exactly 1 Activity
	hasDescription	exactly 1 xsd:string
	i72:value	i72: Measure

2.6.7. Impact Risk

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." (Source: Impact Management Norms) Stating the riskiness of the impact is important for interpreting the subsequent results. Risk is one of the five dimensions of impact as defined by the Impact Norms.

The following defines the key properties of ImpactRisk and its subclasses:

- forImpactReport: Identifies the ImpactReport (and thus the associated StakeholderOutcome) that the risk is associated with.
- hasLikelihood: Identifies the likelihood that the risk will occur among the options given.
- hasConsequence: Identifies the degree of impact the risk could have.
- hasMitigation: A string that specifies a mitigation plan or references a document.
- hasIdentifier: A unique identifier for this risk.
- hasDescription: A description of the risk.

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

Class	Property	Value Restriction
ImpactRisk	forImpactReport	only ImpactReport
	hasLikelihood	exactly 1 {veryUnlikely, unlikely, likely, veryLikely, lowRisk, mediumRisk, highRisk}
	hasConsequence	exactly 1 {minimal, average, severe}
	hasMitigation	exactly 1 xsd:string
	hasDescription	exactly 1 xsd:string
	hasIdentifier	exactly 1 xsd:string
EvidenceRisk	rdfs:subClassOf	Only ImpactRisk
	hasIdentifier	value “Evidence Risk”
ExternalRisk	rdfs:subClassOf	Only ImpactRisk
	hasIdentifier	value “External Risk”
StakeholderParticipationRisk	rdfs:subClassOf	Only ImpactRisk
	hasIdentifier	value “StakeholderParticipation Risk”
DropOffRisk	rdfs:subClassOf	Only ImpactRisk
	hasIdentifier	value “DropOff Risk”
EfficiencyRisk	rdfs:subClassOf	Only ImpactRisk
	hasIdentifier	value “Efficiency Risk”
ExecutionRisk	rdfs:subClassOf	Only ImpactRisk
	hasIdentifier	value “Execution Risk”
AlignmentRisk	rdfs:subClassOf	Only ImpactRisk
	hasIdentifier	value “Alignment Risk”
EnduranceRisk	rdfs:subClassOf	Only ImpactRisk
	hasIdentifier	value “Endurance Risk”
UnexpectedImpactRisk	rdfs:subClassOf	Only ImpactRisk
	hasIdentifier	value “UnexpectedImpact Risk”

2.7. Stakeholder and Stakeholder Characteristics

A Stakeholder either benefits from the services of an organization or contributes to them. The Stakeholder class identifies what activities they perform using the *performs* property, and 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 beneficial and contributing stakeholders. A BeneficialStakeholder is a stakeholder that benefits from a logic model’s outcome. A ContributorStakeholder is a stakeholder that contributes input to a service that can produce outcomes. These are explicitly defined as separate classes because some impact measurement approaches, such as SROI, distinguish between the two.

The following graph (Figure 9) depicts the main classes and properties of a Stakeholder:

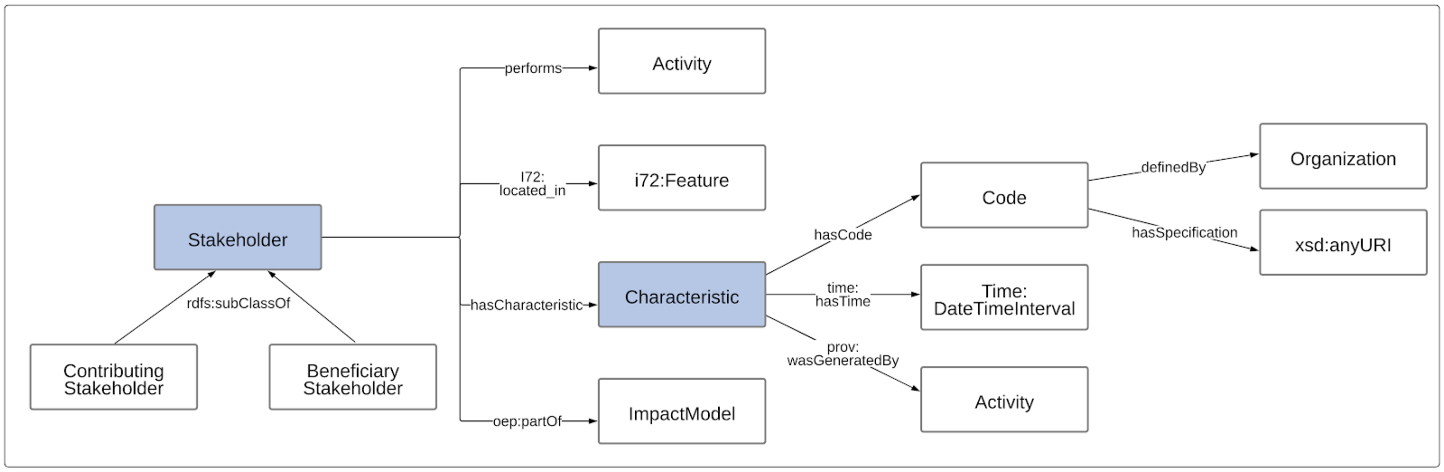


Figure 9: Stakeholder Pattern

2.7.1. *Stakeholder*

A stakeholder may be a person, an organization or a feature. Stakeholders may be described as either Contributing or Beneficial stakeholders, as suggested by Social Value International. Stakeholders may also be identified by their roles, as specified by the Impact Management Norms: Customers, Employees, Communities, Suppliers, and Planet. Stakeholder contains the following properties:

- hasName: A title for the stakeholder as a string.
- hasDescription: A general description of the stakeholder as a string.
- hasCatchmentArea: Specifies the regional span of the stakeholders.
- hasCharacteristic: 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:partOf: Links the Impact Model that the stakeholder is being specified for.

Class	Property	Value Restriction
Stakeholder	rdfs:subClassOf	only Person or Organization or gn:Feature
	hasName	exactly 1 xsd:string
	hasDescription	exactly 1 xsd:string
	hasCatchmentArea	exactly 1 {“local”, “provincial”, “national”, “multinational”, “global” }
	hasCharacteristic	only Characteristic
	performs	some Activity
	i72:located_in	only i72:Feature
	oep:partOf	only ImpactModel
BeneficialStakeholder	rdfs:subClassOf	Stakeholder
	benefitsFrom	min 1 Outcome
	org:hasRole	only {Client, Community, Employee, Planet, Supplier}
ContributingStakeholder	rdfs:subClassOf	Stakeholder
	contributes	min 1 Input

2.7.2. *Characteristic*

There are a myriad of characteristics that social purpose organizations might use to identify a beneficial stakeholder. There are characteristics that the stakeholder must have (i.e., requirements) to be eligible for their services. There are also characteristics that the organization might track to learn more about which people are accessing their services. Common stakeholder characteristics are gender, age, ethnicity, income, geographic location, and disability.

The specification of stakeholder characteristics is often theme dependent. For example, social purpose organizations working to end homelessness often define stakeholders by characteristics such as length (of time) and frequency of homelessness, and location of homelessness (e.g., street, shelter, friend’s home). These characteristics are used to determine which services are relevant. Social purpose organizations are often asked to track specific characteristics for different funders and partnerships.

Listing a set of properties, the approach taken by vocabularies such as FOAF[1] and Schema.org[2], is insufficient for the Common Impact Data Standard for a number of reasons:

- The possible set of properties associated with a Stakeholder is enormous when taking the union across themes and sources, leading to an overloading of the concept.
- The plethora of properties across themes leads to ambiguous and overlapping interpretations.

[1] http://xmlns.com/foaf/spec/
[2] https://schema.org/

- The temporal aspect of a property is ignored, i.e., over what period of time is the property valid (see Katsumi & Fox (2019) for one possible approach to dealing with this).
- The causal aspect of a property is ignored, i.e., what led to the person having the property.

The Characteristic class is used to define a characteristic of a Stakeholder. It has a hasCode property enabling the reuse of defined characteristic taxonomies, and has the following properties:

- forStakeholder: Identifies the Stakeholder affected.
- hasCode: Links to zero or more Codes that are externally defined taxonomies of stakeholder characteristics.
- hasName: Specifies a name for the characteristic as a string. Can be inferred from hasCode.
- hasValue: Specifies a value for the characteristic as a string. Can be inferred from hasCode.
- time:hasTime: The time interval that the Stakeholder has the characteristic.
- prov:wasGeneratedBy: Identifies the activities that led to the Stakeholder having the characteristic.

Class	Property	Value Restriction
Characteristic	forStakeholder	only Stakeholder
	hasCode	Code
	hasName	only xsd:string
	hasValue	exactly 1 xsd:string
	time:hasTime	max 1 time:DateTimeInterval
	prov:wasGeneratedBy	only Activity

2.8. Indicator and IndicatorReport

2.8.1. *Indicator*

In this section, we define the Indicator and IndicatorReport classes. Indicator is a subclass of i72:Indicator (see Section 3.5), which provides properties for units of measure, time and value. cids: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 Outcomes 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 stakeholders and organizations that can read this Indicator.
- hasCode: 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.
- dqv:target: Links to quality policies that the Indicator should conform to.
- hasIdentifier: Unique identifier for the indicator.
- hasName: Specifies the title of the indicator.
- hasDescription: Specifies the indicator's description.
- sch:dateCreated: Is the date that the Indicator was created.

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 Activity
	dcat:dataset	only dcat:Dataset
	hasIndicatorReport (inverse forIndicator)	only IndicatorReport
	hasAccess	only cids:Organization or Stakeholder
	hasCode	only Code
	dqv:target	only dqv:QualityPolicy
	hasIdentifier	exactly 1 xsd:string
	hasName	exactly 1 xsd:string
	hasDescription	exactly 1 xsd:string
	sch:dateCreated	exactly 1 xsd:dateTime

2.8.2. *IndicatorReport*

IndicatorReport is used to report the value of an indicator for some time interval. In addition to the value of the indicator, it reports on when it was generated, the activity used to generate it, the datasets used, and the quality of the value reported using the DQV vocabulary (https://www.w3.org/TR/vocab-dqv/). 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: Specifies a single measure of the result value.
- 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.
- time:hasTime: 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 stakeholders and organizations that can read this Indicator report.
- hasName: Specifies the IndicatorReport's name or title of the indicator.
- 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 Activity
	dcat:dataset	only dcat:Dataset
	time:hasTime	exactly 1 time:DateTimeInterval
	dqv:hasQualityAnnotation	only dqv:QualityAnnotation
	dqv:conformsTo	only dcterms:Standard
	dqv:hasQualityMeasurement	only dqv:QualityMeasurement
	hasAccess	only Organization or Stakeholder
	hasComment	only xsd:string
	hasName	exactly 1 xsd:string
	sch:dateCreated	exactly 1 xsd:dateTime

2.9. 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 mobile services that provide food and clothing to those that experience homelessness, and a training service that provides basic skills for those that experience homelessness. A Program has a set of Stakeholders that may contribute or benefit.

- hasService: Identifies the Services that make up the Program.
- hasName: Identifies the name of the Program.
- hasService: Identifies the Services that make up the Program.
- hasDescription: A description of 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
	hasName	exactly 1 xsd:string
	hasDescription	exactly 1 xsd:string
	hasService	only Service
	hasOutcome	only Outcome
	hasContributingStakeholder	only ContributingStakeholder
	hasBeneficialStakeholder	only BeneficialStakeholder
	hasInput	only Input
	hasOutput	only Output

2.10. Service

A Program is composed of one or more Services. As described above, a poverty reduction program can have many services, with each service comprised of different activities, Inputs, Outputs, and Outcomes. A Service is a subclass of Activity and can be related to one or more Programs.

- oep:partOf: Identifies the Impact Model it is a component of.
- hasName: Identifies the name of the Service.
- hasDescription: A description of the Service.
- act:hasSubActivity: Identifies the Activities 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.
- beneficialSizeStart: Number of beneficial stakeholders at the beginning of the service time interval.
- beneficialSizeEnd: Number of beneficial stakeholders at the end of the service time interval.
- time:hasTime: Time interval over which the service is provided.

Class	Property	Value Restriction
Service	rdfs:subClassOf	Activity
	oep:partOf	exactly 1 ImpactModel
	hasName	exactly 1 xsd:string
	hasDescription	exactly 1 xsd:string
	act:hasSubActivity	only Activity
	hasInput	only Input
	hasOutput	only Output
	hasOutcome	only Outcome
	hasContributingStakeholder	only ContributingStakeholder
	hasBeneficialStakeholder	only BeneficialStakeholder
	beneficialSizeStart	exactly 1 xsd:positiveInteger
	beneficialSizeEnd	exactly 1 xsd:positiveInteger
	time:hasTime	exactly 1 time:DateTImeInterval

2.11. Activity

Activity defines the actions performed by an organization and/or contributing stakeholder to implement a Service. The Common Impact Data Standard”s Activity class is defined to be a subclass of the TOVE Activity (outlined in Section 3.7), 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 than 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.
- oep:partOf: Identifies the Impact Model it is a component of.
- 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.
- hasName: Specifies a name or title for the activity.
- hasDescription: Specifies a description of the activity.
- hasIdentiifier: Specifies a unique identifier for the activity.

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 Code
	act:subActivityOf	only (Service or Activity)
	hasName	only xsd:string
	hasDescription	only xsd:string
	hasIdentifier	exactly 1 xsd:string

2.12. 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.

The Input Class contains the following properties:

- 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.
- time:hasTime: Specifies the time interval over which the Input is provided.
- hasName: Identifies the name of the Input.
- hasDescription: A description of the Input.

Subclasses of Input define properties specific to that subclass.

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
	time:hasTime	only time:DateTimeInterval
	hasName	exactly 1 xsd:string
	hasDescription	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

2.13. Output

An Output is “the direct result of an entity’s activities (e.g. wages paid, hours of training provided, or products and services sold). It may also include changes resulting from the entity’s actions or decisions which are relevant to achieving outcomes.” (Source: UN SDG Impact Standards Glossary). Outputs represent a quantitative summary of an activity. For example, if the activity is “we provide training,” the output is “we trained 50 people to NVQ level 3” (Source: Social Value International, 2012). An example of a production output would be “we produce 100 meals for people experiencing homelessness”. An output represents what has been produced and the quantity. (Note: many impact models illustrate outputs leading to outcomes, but a more precise representation is that activities lead to outcomes. Outputs describe the quantity of activity.)

- eof:partOf: Specifies the impact models this Output is part of.
- forActivity: Identifies the Activity or Service that produces the Output.
- i72:value: Identifies the 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.
- hasName: Identifies the name of the Output.

- hasDescription: A description of the Output.

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
	org:hasName	exactly 1 xsd:string
	hasDescription	exactly 1 xsd:string

3. Foundational Ontologies

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

3.3. Date/Time – OWL-Time:2020

3.3.1. Introduction

Time is both absolute and relative. To understand impact data, it is often necessary to know at what time something occurred and 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" (https://www.w3.org/TR/owl-time/).

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

A ProperInterval may overlap, contain, or in some other way be related to another defined ProperInterval. These relationships can be captured using the ProperInterval properties that start at time:intervalBefore. [3]

Figure 10 depicts the core classes that comprise OWL-Time.

[3] 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.

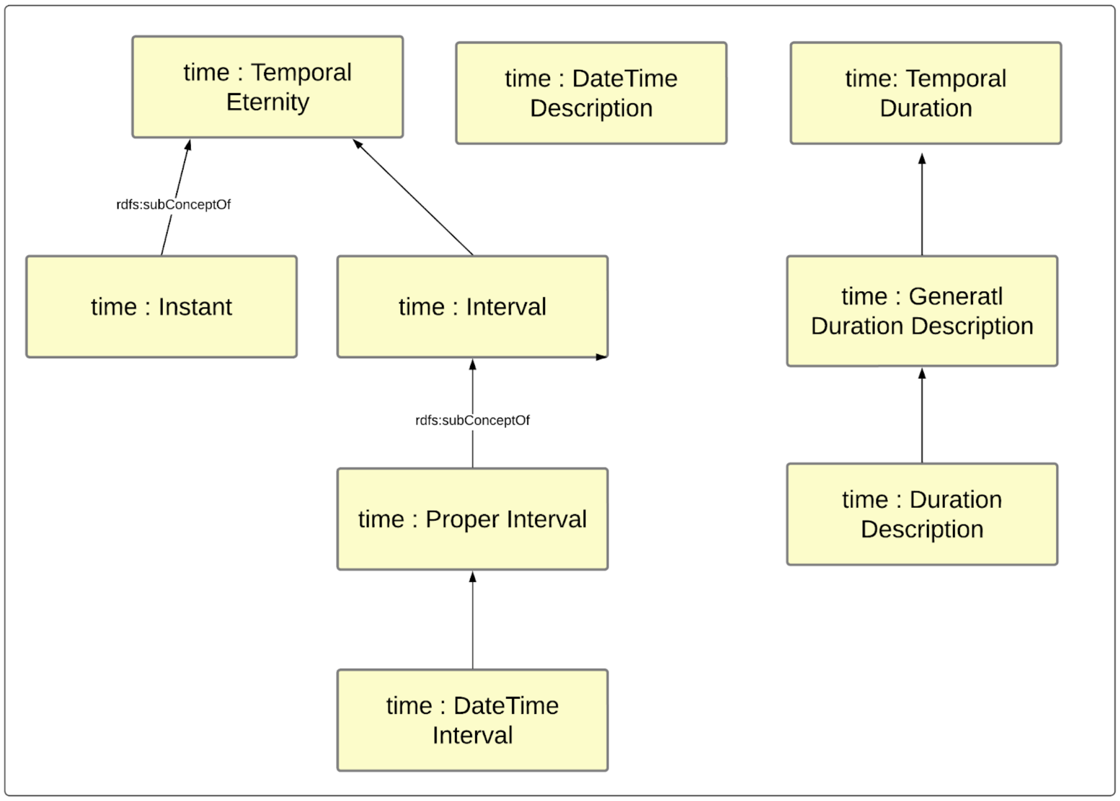


Figure 10: Time Concepts. Figure 10 depicts the relationships that TemporalEntity has with the other classes.

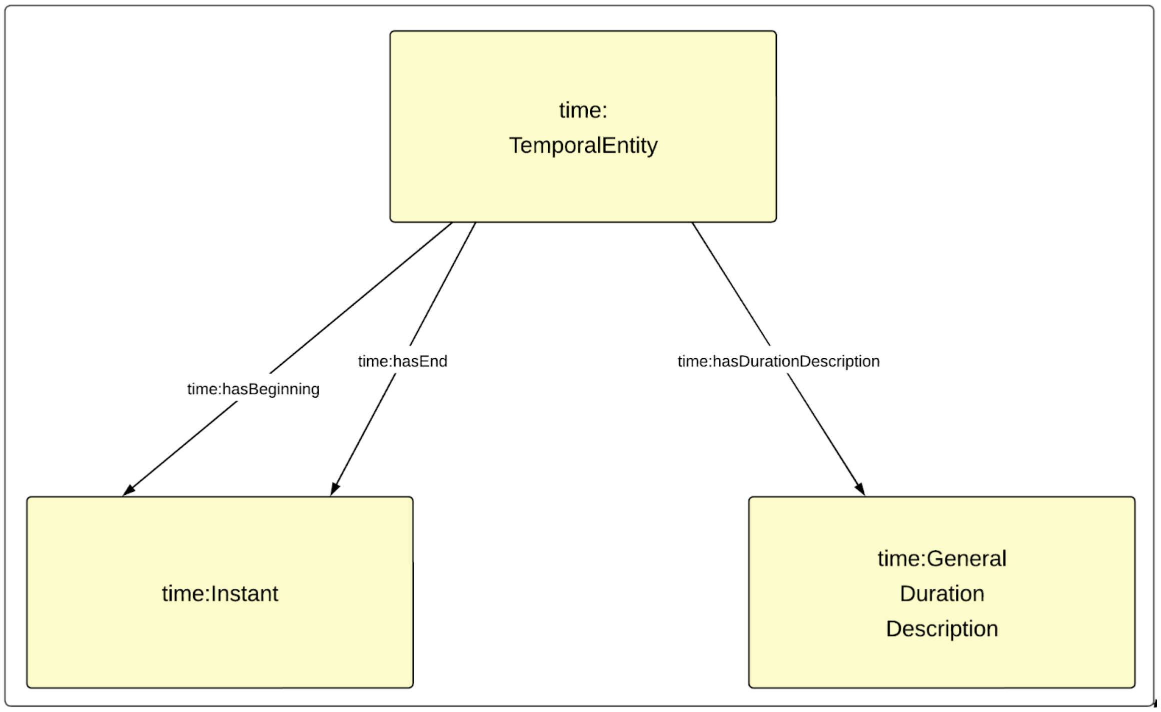

time: TemporalEntity
time:hasBeginning
time:hasEnd
time:hasDurationDescription
time:Instant
time:General Duration Description

Figure 11: TemporalEntity Relationships

3.3.3. Formal Specification

The following defines the basic classes and their properties.

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:DateTimeInterval	rdfs:subClassOf	time:ProperInterval
	time:hasBeginning	time:DateTimeDescription
	time:hasEnd	time: DateTimeDescription
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)

3.4. Address – tove/icontact

3.4.1. Introduction

Addresses across the globe vary in the types of information they include. For example, a British 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 can be accessed at http://ontology.eil.utoronto.ca/tove/icontact.owl.

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

3.5. 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	{ic:mainline, ic:faxPhone, ic:cellPhone}

3.6. Measurement – ISO/IEC 21972:2020

3.6.1. 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 ensure 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, they are expressed in the same scale. Consider an advocacy group that is working to end homelessness. They use data from two different sources to track the percentage of the homeless that are male. To do this, they need to know that the data of homeless men and total homeless are of the same scale (for example: hundreds versus 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 the definition of indicators.

3.6.2. 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 12 depicts the basic classes of the measurement ontology. There are three main classes:

- A Quantity that denotes what is being measured (e.g., the 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'." (Rijgersberg et al., 2013).
- 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, Male 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 'Male homeless ratio measure'.

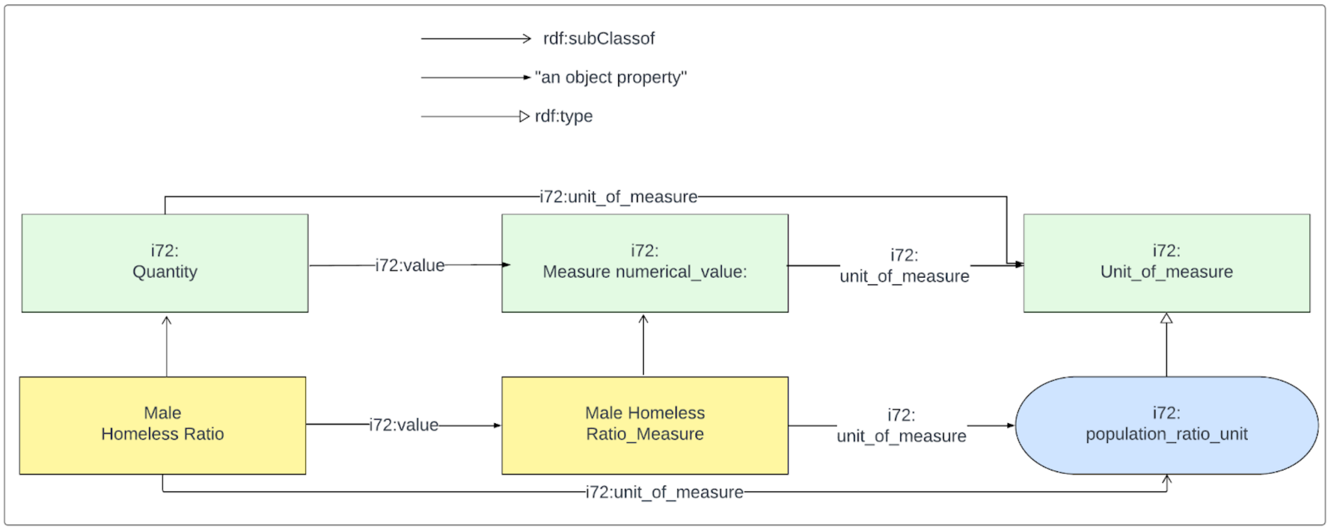


Figure 12: Basic Measurement Classes

The concept of a Quantity is common across many standards. 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 13:

- 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 more than one Unit_of_measure. For example, speed would be an instance of a Unit_division where a singular (e.g., metre) or multiple unit (e.g., kilometer) would be divided by time (e.g., hour).

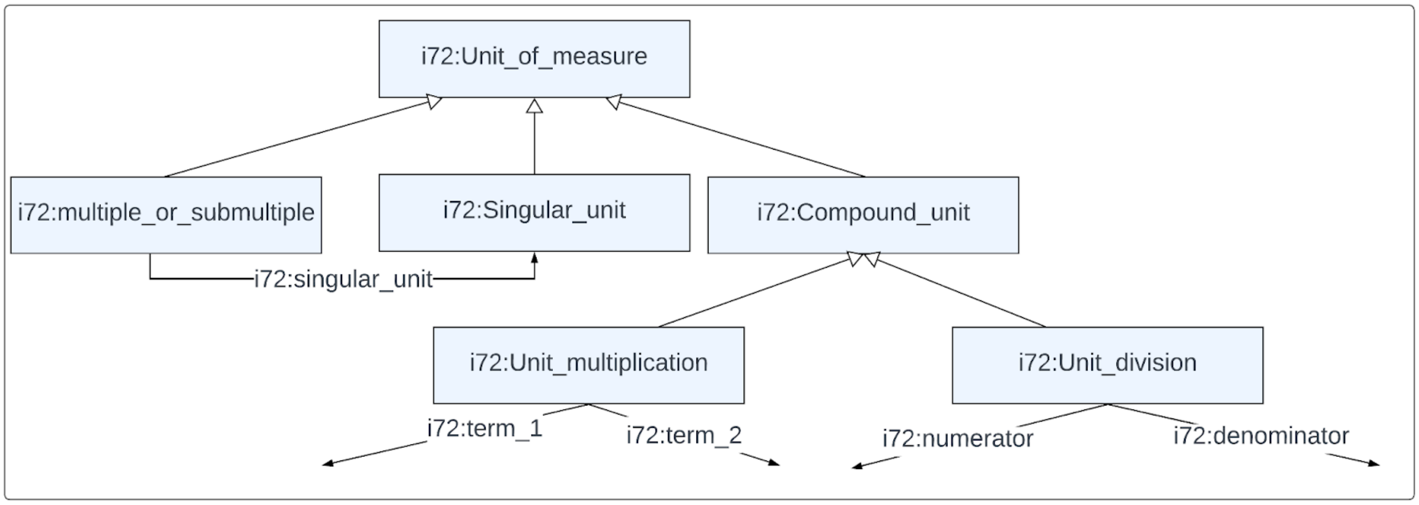


Figure 13: Unit of Measure Taxonomy

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[4], a zero point can be defined. The measurement scale taxonomy is illustrated in Figure 14.

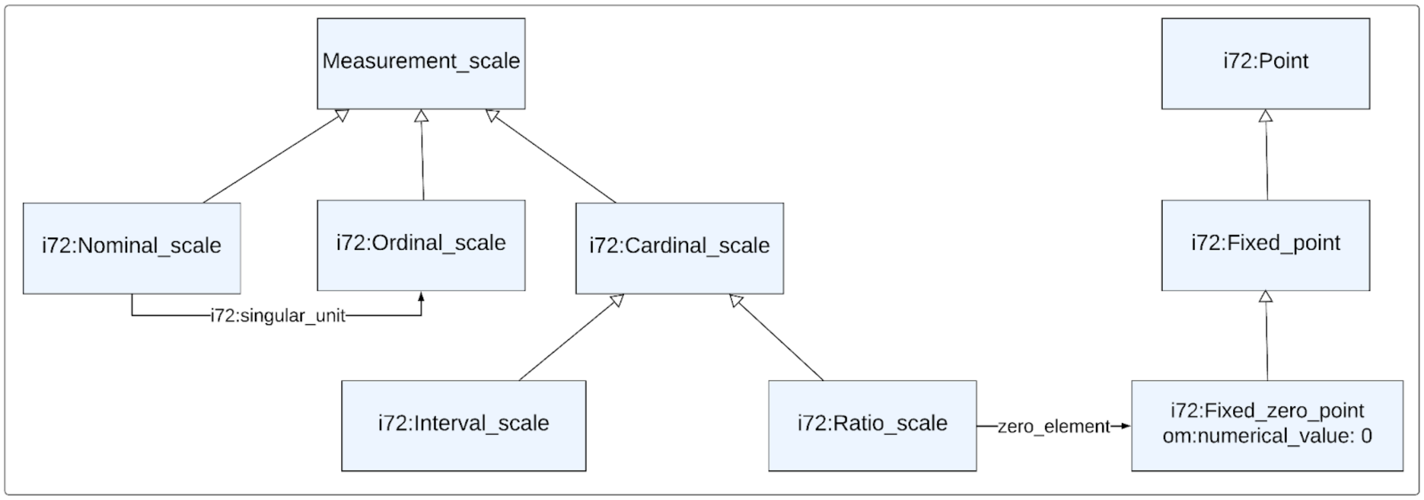


Figure 14: Measurement Scale Taxonomy

[4] "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" (Source: http://www.statisticshowto.com/ratio-scale/).

3.6.3. 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)”.

3.7. Indicator – ISO/IEC 21972:2020

3.7.1. Introduction

Indicators are used to measure the Outcomes of social purpose organizations. A challenge for social purpose organizations is to define Indicators in ways 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. 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.

3.7.2. Core Classes and Properties

The Common Impact Data Standard 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 15 depicts the specification of the Cardinality_unit.

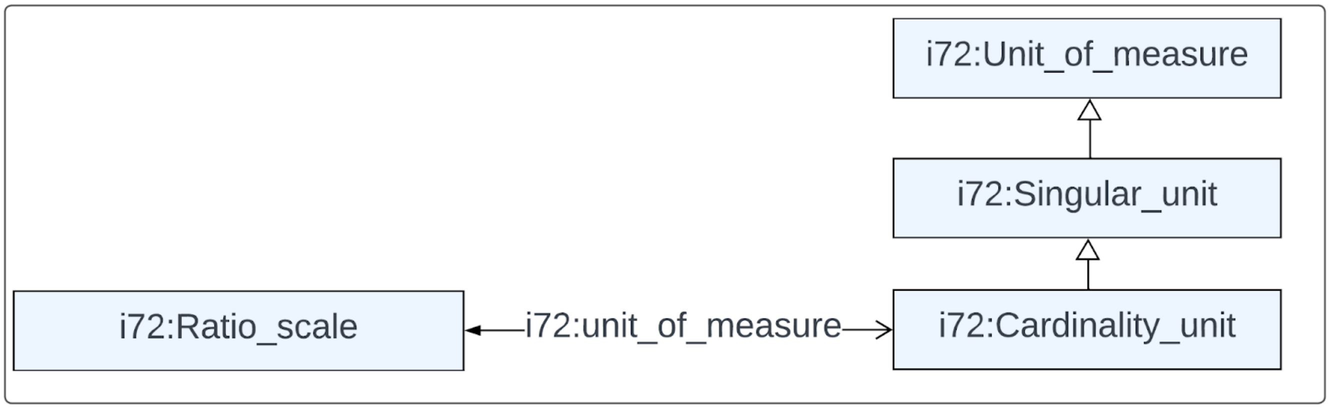


Figure 15: Cardinality_unit Definition

In Figure 16, 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.

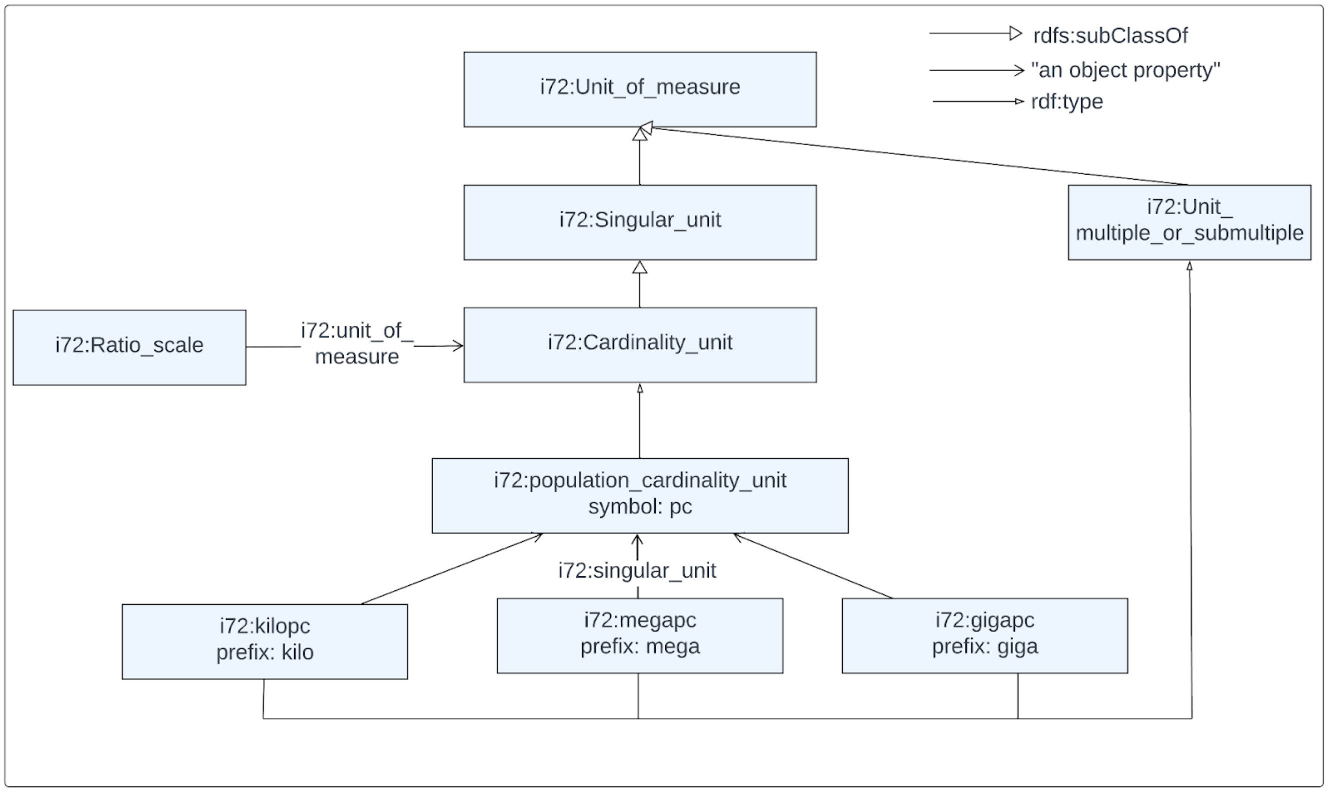


Figure 16: population_cardinality_unit Definition

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 16 above). 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 17 provides the unit of measures for populations (pc) and population ratios (pc/pc).

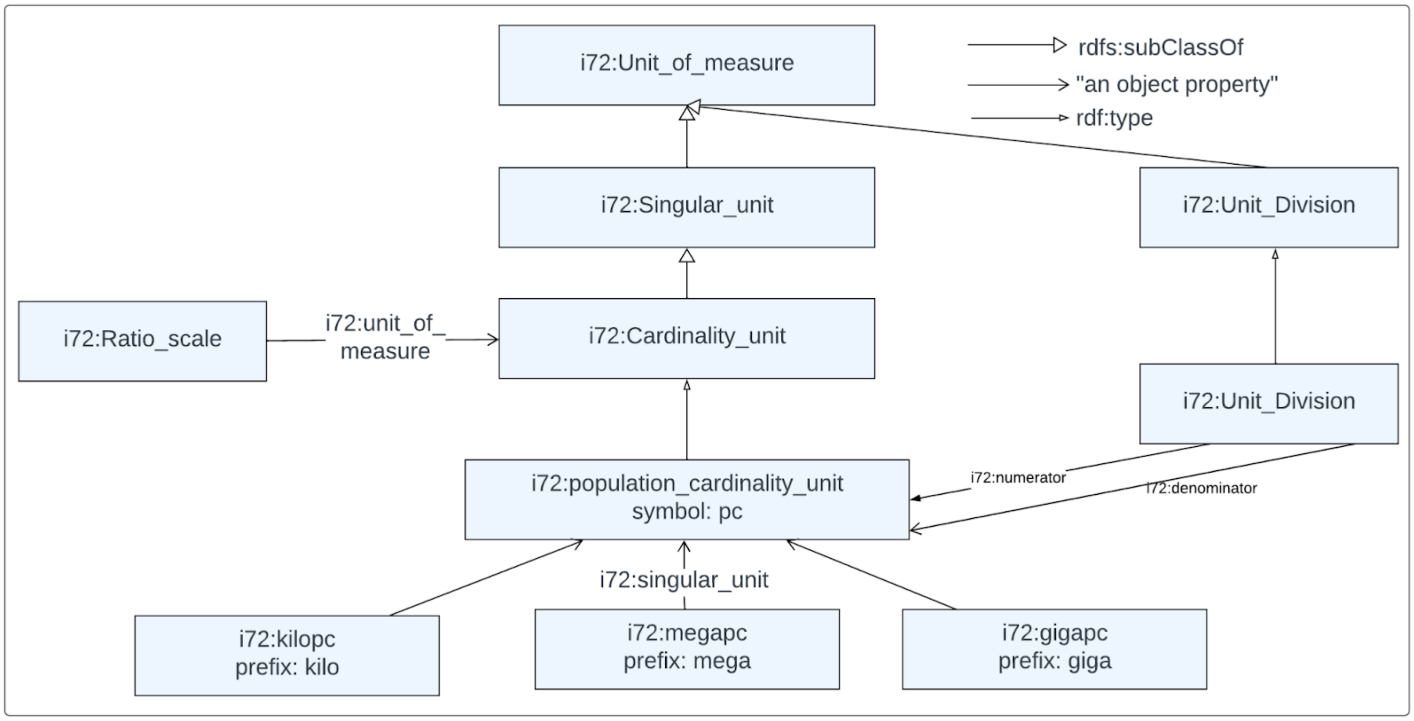


Figure 17: Depicts the population ratio unit definition

3.7.3. 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:denominator	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. 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 18 depicts the translation of the Indicator "Average Number of Skills each Job Seeker gained", the indicator ontology, which is based on ISO 21972.

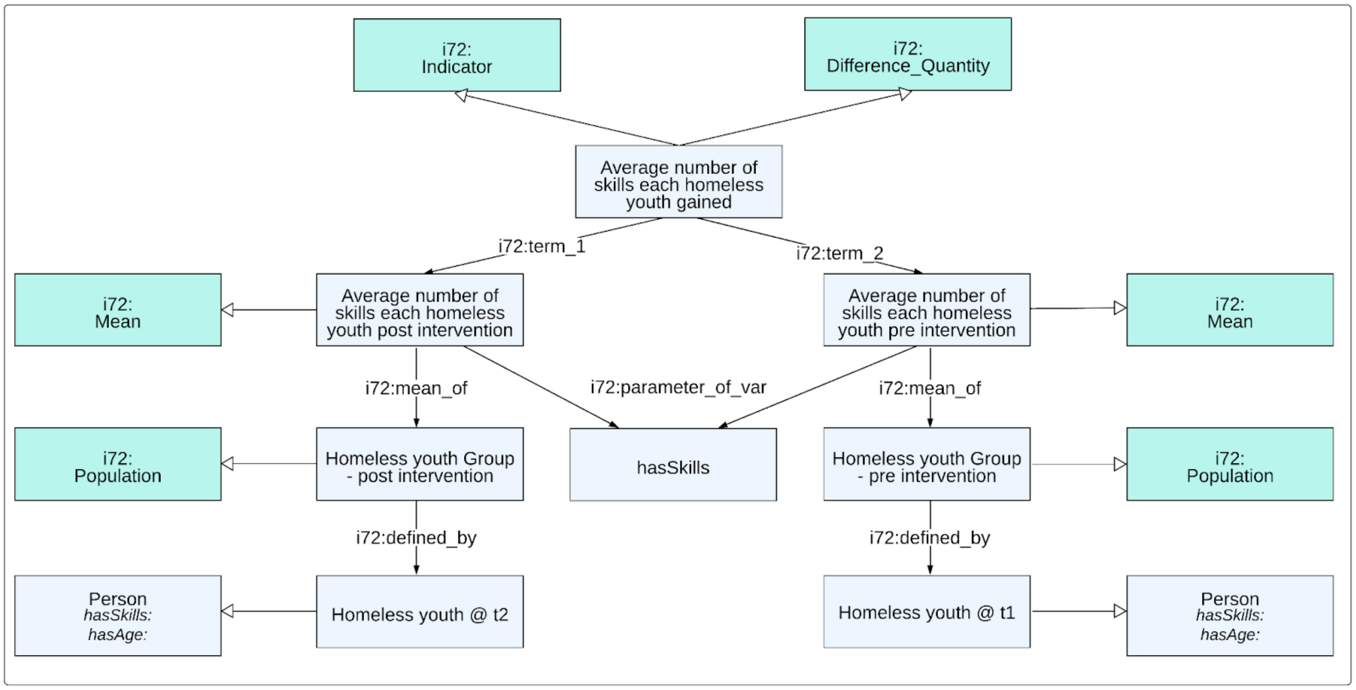

i72: Indicator
i72: Difference_Quantity
Average number of skills each homeless youth gained
i72:term_1
i72:term_2
i72: Mean
Average number of skills each homeless youth post intervention
Average number of skills each homeless youth pre intervention
i72: Mean
i72:mean_of
i72:parameter_of_var
i72:mean_of
i72: Population
Homeless youth Group - post intervention
hasSkills
Homeless youth Group - pre intervention
i72: Population
i72:defined_by
i72:defined_by
Person hasSkills: hasAge:
Homeless youth @ t2
Homeless youth @ t1
Person hasSkills: hasAge:

Figure 18: Indicator Pattern

3.8. Person

3.8.1. Introduction

The 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 Common Approach Foundation Ontology. CWRC Ontology Specification is accessible at http://sparql.cwrc.ca/ontology/cwrc.html.

3.8.2. 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.

3.8.3. 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
	cwrc:parentOf	Only Person
	cwrc:hasGender	only cwrc:Gender
	cwrc:hasEthnicity	only cwrc:Ethnicity
	cwrc:hasReligion	only cwrc:Religion
	hasOccupation	only xsd:string
	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

3.9. Activity – tove/activity

3.9.1. 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.

3.9.2. Classes and Properties

Action is represented by the combination of an activity and its corresponding enabling and caused states (Figure 19). 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.

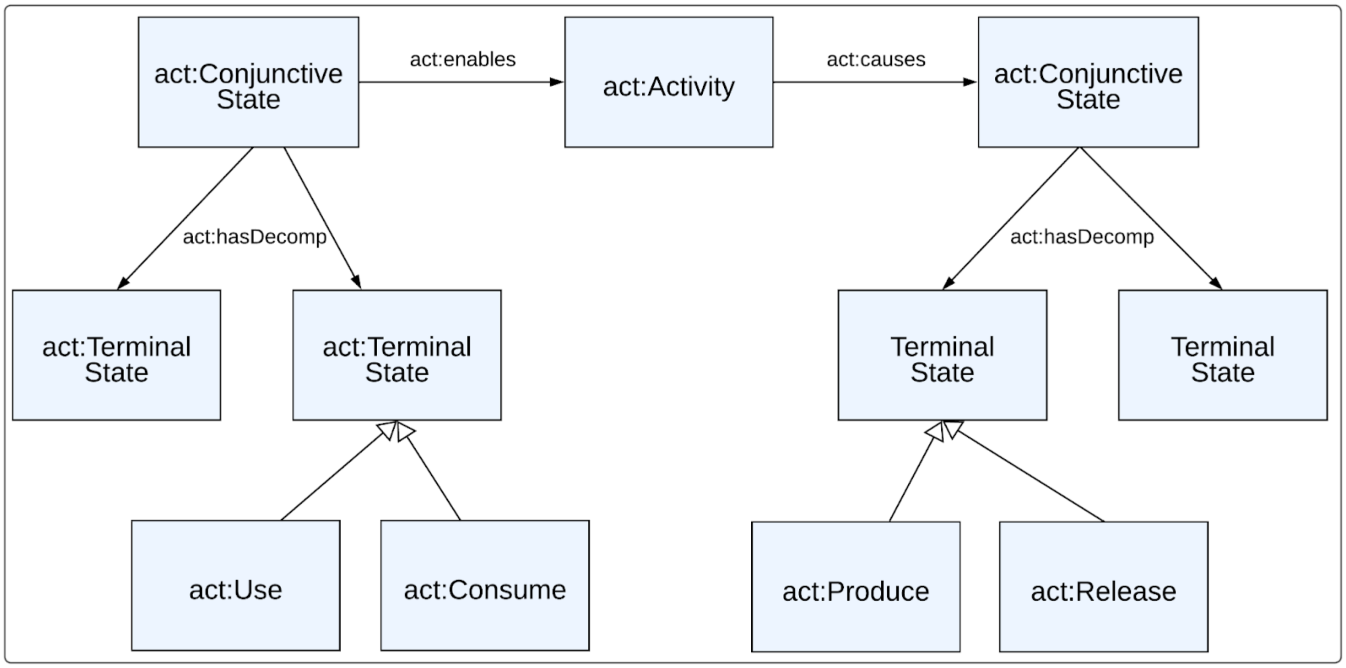


Figure 19: Activity-State Cluster

The status of an activity is reflected in an attribute called status. We define the theme 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 to an activity by act:nabledby relation specifies what has to be true in order for the activity to be performed. The state tree linked to an activity by 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 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 theme of a state's status as a set of linguistic constants. For example, the theme 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.

3.9.3. 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: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:disjointWith	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

3.10. Location

The ontology for representing location information shall conform to the vocabulary specified in OGC 11-052r4 (GeoSPARQL). To capture generic spatial features requires concepts of location, but also concepts of geometry in order to describe shapes that are more complex than a single point in space. In addition, there is a need to be able to describe the spatial relationship between various features (e.g. containment, overlap). The GeoSPARQL Ontology is used in the Location Pattern to achieve this. It is included in its entirety with the prefix 'geo'.

3.10.1. Classes and properties

The key classes and properties are formalized in Table 3 and Table 4, respectively. In this subclause, a subset of the GeoSPARQL ontology is replicated and specialized:

- **Feature**: represents some area in space. Features are distinguished by their geometric shape as well as their geographic location. Core properties are:
    - **hasGeometry**: specifies the shape that defines the location of the Feature in order to capture quantitative spatial information.
- **Geometry**: represents a shape. Various types (subclasses) of Geometry are defined in the GeoSPARQL ontology including the classes: Point, Polygon, and Curve. Core properties are:
    - **asWKT**: specifies the well-known text encoding of a given geometry. The default reference system for the coordinate values is assumed to be WGS84. GeoSPARQL supports the identification of alternate reference systems, captured as IRIs and concatenated with the coordinates.

In addition, the pattern specifies the following generic properties to support the reference of locations by other classes:

- **hasLocation**: captures the relationship between objects and the spatial locations they occupy.
- **associatedLocation**: introduced to capture the association of a given object with a particular location. For example, a train station can occupy a fairly large spatial location but be associated with a particular point.

3.10.2. Formalization

Class	Property	Value restriction
Feature	rdfs:subClassOf	geo:Feature
	hasGeometry	exactly 1 geo:Geometry
Geometry	rdfs:subClassOf	geo:Geometry
	asWKT	exactly 1 geo:wktLiteral

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., 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).

Poblet, M., Casanovas, P., & Rodríguez-Doncel, V. (2019). Introduction to Linked Data. In *Linked Democracy* (pp. 1-25). Springer, Cham. https://link.springer.com/chapter/10.1007/978-3-030-13363-4_1

Rijgersberg, H., Wigham, M., and Top, J.L., (2011), "How Semantics can Improve Engineering Processes: A Case of Units of Measure and Quantities", Advanced Engineering Informatics, Vol. 25, pp. 276-287.

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

Appendices

Appendix 1: JSON-LD Example at Basic Tier with annotations

In this section we show how the Common Impact Data Standard can be used as an interchange format among SPOs, Impact Measurement platform providers and other repositories. We show an example of the Basic Tier for a fictitious organization Affordable House Inc. using the JSON-LD syntax[5].

For more detail on the various tiers of alignment, please refer to the Alignment Tiers guide.

This example depicts an instance of the following CIDS classes:

- cids:Organization
- cids:Outcome
- cids:Indicator
- cids:IndicatorReport

The Organization instance contains a blank node instance of org:OrganizationID (see annotation for details). Similarly, the Indicator Report shows how to use blank node instances of more complex structures such as time:DateTimeInterval and i72:Measure.

The following requirements must be satisfied in order to use the representation properly:

1. An Organization must have a URI/IRI that uniquely identifies the organization. The Common Approach can provide one if data is to be stored in the Common Approach Repository.
2. Each instance of a class that is not embedded as a blank node must have an “@id” property that provides a unique identifier.
   Note: If there is no URI but the node needs to be referenced multiple times, you can generate an internal @id with a blank prefix e.g. “_:n1”, “_:b3”, “_:blank48”. This id can be used in the same way as a URI to link instances in the JSON-LD file. See below for an example of how ImpactModel links to Indicators with its URI.

[5] We chose JSON-LD for two reasons: 1) JSON is easier to read than a syntax like XML; 2) JSON-LD is designed for linked data, which the CIDS ontology conforms to. The ability to specify namespaces and types is essential to the proper use of CIDS. Turtle is a good alternative, but less widely adopted. Information about JSON-LD, tutorials and a validator (called Playground) can be found at: https://json-ld.org/.

```
<script type="application/ld+json">
[
{   "@context" :
"http://ontology.eil.utoronto.ca/cids/contexts/CIDSOrganizationContex
t.json" ,
    "@type" : "cids:Organization" ,
    "@id"          : "http://AffordableHouse.org" ,
    "hasID" : { "@context" :
"http://ontology.eil.utoronto.ca/CIDS/contexts/OrganizationIDContext.j
son" ,
                    "@type" : "org:OrganizationID" ,
                    "issuedBy" :
"http://https://www.canada.ca/en/revenue-agency.html" ,
                    "identifier" : "12345678"} ,
  "hasLegalName" : "Affordable House Inc."
} ,
{   "@context" :
"http://ontology.eil.utoronto.ca/cids/contexts/OutcomeContext.json" ,

    "@type" : "cids:Outcome" ,
    "@id"          : "http://AffordableHouse.org/outcome1" ,
    "hasName": "Tenant accommodation",
    "hasDescription": "Tenants have accommodation that they want",
    "dateCreated": "2021-03-21"
} ,

{   "@context" :
"http://ontology.eil.utoronto.ca/cids/contexts/IndicatorContext.json" ,

    "@type" : "cids:Indicator" ,
     "@id" : "http://AffordableHouse.org/indicator1" ,
    "name": "Length of Tenure",
    "description": "the percentage of TMs who have lived in Affordable
House accommodation for more than 5 years"
} ,

{   "@context" :
"http://ontology.eil.utoronto.ca/cids/contexts/IndicatorReportContext.j
son" ,
    "@type" : "cids:IndicatorReport" ,
    "@id"          : "http://AffordableHouse.org/indicatorReport1" ,
```

This is an example of a blank node that is only used by one other node. It provides the org:OrganizationID object that links the Organization to an ID that is provided by an external issuer. In this case, the identifier is a business number issued by the Canada Revenue Agency. If it were needed by other nodes, it would be better to assign it a 'blank node ID' and define it in its own context.

hasLegalName is a required field for all Organizations. Each Organization should have exactly one instance of this property, given as a string. See table for details.

This is NOT a blank node. This URI points to an indicator defined by AffordableHouse. It should have an associated definition available at this address. The id provided here can be referenced by other nodes in the graph. See 'forIndicator' in the IndicatorReport object below.

```
    "hasTimeInterval" : { "@context" :
"http://ontology.eil.utoronto.ca/CIDS/contexts/TimeContext.json" ,
                        "@type" : "time:DateTimeInterval" ,
                        "hasBeginning" : { "@type" :
"time:DateTimeDescription" ,
                                          "year" : "2020" ,
                                          "month" : "1" ,
                                          "day" : "1"
                        } ,
                        "hasEnd" : { "@type" :
"time:DateTimeDescription" ,
                                    "year" : "2020" ,
                                    "month" : "12" ,
                                    "day" : "31"
                        }
                      } ,
    "forOrganization" : "http://AffordableHouse.org" ,
    "forIndicator" : "http://AffordableHouse.org/indicator1" ,
    "value" : { "@context" :
"http://ontology.eil.utoronto.ca/CIDS/contexts/ISO21972Context.json"
,
            "@type" : "i72:Measure" ,
            "numerical_value" : "75"
          } ,
     "dateCreated" : "2021-03-21"
}
]


</script>
```

This property creates a link to the Indicator1 ('Length of Tenure') object defined above.

This blank node is a i72:Measure object nested in and belonging to the indicatorReport1 object.

Through the above id link, this IndicatorReport connects a value of 75 to the 'percentage of TMs who have lived in Affordable House accommodation for more than 5 years.' indicator

Appendix 2: Prefixes used in this document and the connected URI

Prefix	URI
act	http://ontology.eil.utoronto.ca/tove/activity#
crwc	http://sparql.cwrc.ca/ontologies/cwrc#
cids	http://ontology.commonapproach.org/owl/cids_v2.1
dcat	http://www.w3.org/ns/dcat#
dqv	http://www.w3.org/ns/dqv#
foaf	http://xmlns.com/foaf/0.1/
gn	https://www.geonames.org/ontology#
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#
owl	http://www.w3.org/2002/07/owl#
rel	http://purl.org/vocab/relationship/
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#