iCity Ontology Version 1.2 Report

Design Documentation

Megan Katsumi
Post-Doctoral Fellow
Enterprise Integration Lab
Mechanical and Industrial Engineering
University of Toronto

Mark Fox
Distinguished Professor of Urban Systems Engineering
Professor of Industrial Engineering and Computer Science
Director of the Centre for Social Services Engineering
University of Toronto

Last Updated: April 8, 2019

UNIVERSITY OF TORONTO
FACULTY OF APPLIED SCIENCE & ENGINEERING
Transportation Research Institute

UTTRI

1 Purpose

The purpose of this document is to present the first release of the iCity ontology. This document provides a concrete outline of the concepts defined in the ontology; its purpose is to provide a point of reference to facilitate discussion and feedback on of the ontology, and to facilitate its implementation across the iCity projects and beyond. This initial release focuses on the identification of the classes and properties that will form the ontology. These classes and properties provide a clear indication of the breadth of the scope of the iCity ontology, and are thus a critical first step in the development of the final artifact. Based on this release, we may begin the process of implementation by capturing relevant information in the language defined by the ontology, and thus providing a shared, commonly understood model of the knowledge being used and generated by the various iCity projects.
HTML documentation is maintained for each ontology, automatically generated using Widoco[1].

2 Scope

The scope of this document is limited to the identification of the vocabulary within the ontology. More specifically, it is restricted to the identification of vocabulary in the Urban System Ontology; at this stage, it does not capture the application-specific concepts of the individual iCity projects. We provide an initial specification of the classes and relationships (properties) to support formalization in OWL 2 (Grau, et al., 2008). Future versions will expand on the depth of these definitions, providing more detailed semantics in a complementary logical language. This document aims only at conveying the vocabulary currently defined in the iCity ontology, implementation of the ontology shall be addressed in a separate iCity report.

3 Outline

The report will begin with an introduction to the role of the ontology within the iCity project. The core concepts pertaining to the characteristics and behaviour of the urban system will then be presented in Section 5. Section 0 identifies directions for future iterations of the ontology; in particular, Section 8.2 outlines top-level concepts required for data collection, simulation, and analysis applications. At the current stage of development, we have not identified any requirements specific to the Visualization application (Theme 3 in the iCity project). It is our understanding that the Theme 3 work will interpret the iCity ontology in order to generate the required visualizations. Should this change in the future, it is likely that an extension to capture the visualization applications will also be required.

4 Role of the Ontology

Given that all of the projects within iCity are situated in the urban domain, and therefore it is not surprising to find many common concepts between them. It stands to reason that some integration between the different applications should be possible. For example, if data is collected about the population, it should be usable by ILUTE and other simulations, but also by the projects developing analysis tools, such as the smart parking application. Unfortunately, there is also ambiguity in how different concepts are used, and in some cases the same concept may be defined differently in different applications. This provides a challenge not only for integration of

[1] https://github.com/dgarijo/Widoco

the iCity applications, but for shareability of results: if the knowledge generated by iCity is not defined sufficiently, it will be difficult for any other researchers to understand and leverage it. The iCity ontology provides a common set of terms with which data can be stored and accessed. The ontology will resolve any ambiguities and disagreements between terms by defining a common set of concepts that completely captures the domain, with agreed-upon definitions. In the case that two applications attribute a different meaning to the same term, the result will be two distinct terms with distinct, precisely defined meanings. In this way we can recognize these differences and clearly identify the relationships between different concepts. The ontology will be used to organize and describe data within the iCity project. It may also be used as means of publishing or sharing data with the research community.

The resulting artifact, often referred to as the *knowledge base* will take the form of a triple-store(s)[2], created by mapping data from the iCity applications to the agreed-upon terminology defined in the iCity ontology. The architecture for the ontology's implementation in the context of the iCity project, is illustrated in Figure 1.

The precise and formal nature of the ontology will support the use of services such as inference and data validation. Based on the definitions, we may be able to infer new information that was not originally part of the knowledge base. Data validation is supported as a result of the consistency-checking mechanism. We also hope that identification of relationships may serve to uncover synergies between the projects, by illustrating how data from one project may serve to inform the work of another.

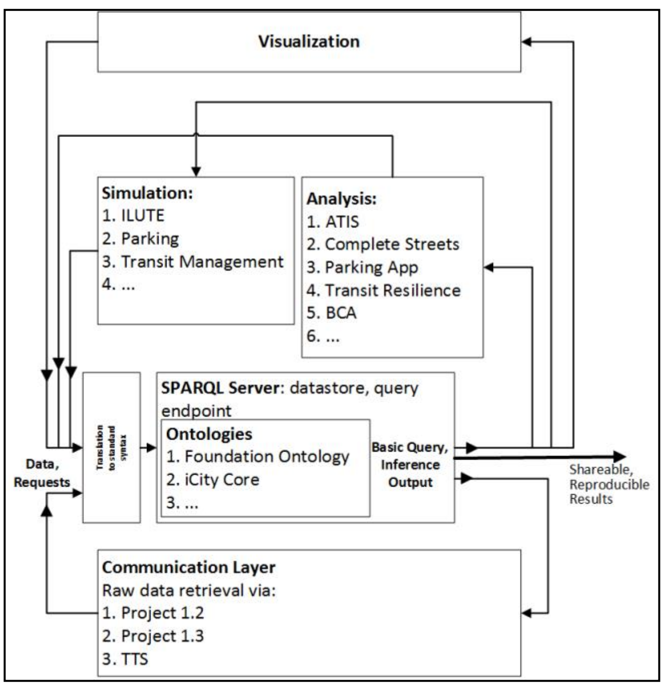


Figure 1: iCity Knowledge Base Architecture

[2] Note that the data may not in fact be stored in the triple-store, but maintained in the application's own database, with mappings from the ontology to the database performed on-the-fly, as required.

The sections that follow introduce the core ontology required to capture the iCity projects, in particular, to define the urban system. Beyond this, the iCity ontology may be implemented to support specific applications. Examples of this are discussed in further detail in Section 0.

5 Summary of Changes from Previous Version

The “iCity-“ prefix was removed from all ontology filenames and IRIs in order to improve readability and convey generality. All other changes are summarized by ontology below.

Activity ontology:

- revised representation such that an activity is a class of occurrences (activities and occurrences are not separate entities); removal of ActivityOccurrence, definition of State instead of StateType.

Transportation Network ontology:

- added Link and LinkPD classes to serve as “containers” for multiple arcs (e.g. vehicle lanes, bicycle lanes, walkways); introduced some additional properties and changed the mode of an Arc from an invariant to variant property.
- Associated Nodes with location information.
- Links and Arcs represent access on some part of the physical infrastructure, which has an associated location. Although Nodes do not represent access in the same way, they are still associated with some physical location. A property was added to specify the *associated* location of a Node. It should be possible to infer which Transportation Complexes (e.g. road segments) meet or contain the node based on the links it is connected to. Future extensions may consider capturing the relationship between the nodes location and the location of the Transportation Complexes accessed by its related links.

Added: iContact.owl Ontology

The notion of addresses is required to represent information about buildings, parking lots, the start and end of trips, and so on. Contact information for individuals and organizations may also be required and captured for some applications.

Added: Calendar/Hours of Operation Ontology:

Beyond the representation of individual timepoints and time intervals, there is often a requirement to reference concepts from a calendar. In particular, the specification of hours of operation (e.g. fora business or transportation network policy) relies on the representation of these concepts, such as the days of the week or times of day. The Calendar Ontology is introduced to define these concepts.

Imported SSN/SOSA Ontology:

Sensor observations are an integral part of ITS operations and research. To capture these sensors and the data they generate, we import the SSN/SOSA ontology [ref].

Units of Measure Ontology:

- Extended and merged with monetary value ontology.

- Updated with new release OM 2.0
- Extend with specializations of quantities, measures, etc, as required by use cases.

Spatial Location ontology:

- Replaced original representation with geoSPARQL terminology, primarily due to it being better supported (geoSPARQL works for simple linked data but also offers the potential for specialized query abilities) and more current.
  Features have geometry (geo:hasGeometry); geometries can be defined as simple features (points, polygons,…); these geometries can be *serialized* as WKT or GML, special purpose datatypes that allow for, e.g. a series of coordinates. Both serializations support the specification of a reference system, therefore (for now) we do not need to extend OM with NAD83 and WGS84.
  The representation of the reference system is not ideal as in the current implementation of geoSPARQL it is appended within the same IRI as the coordinate data, (it is also not clear what code is to be used for NAD83). Future revisions should investigate a possible extension to this representation that will capture the reference system in a more convenient way (while still leveraging the capabilities of geoSPARQL).
  Ideally, in future work we would like to look consider the spatial-location theory in more detail as it geoSPARQL provides a vocabulary and a tool but lacks a complete declarative semantics
- This change impacts many of the other ontologies within iCity (those with spatial information). Each was modified to address this.
- Extended SpatialLoc to include a hasLocation property. This allows us to separate objects from their spatial embodiment as required. Whether an object is related to geosparql:Feature or is a subclass of geosparql:Feature is a foundational ontological decision. In either case, we can describe location of these objects more precisely via the hasGeometry property.

Foundation Ontology: removed

- In the initial design Foundation is imported by all of the domain ontologies, this requires a revision to all of the Urban System ontologies. We observe that this design is not ideal as there may be cases where foundational concepts (e.g. activities or resources) are not used in a particular domain ontology. In such cases, updates to Foundation.owl will result in unnecessary updates to unaffected domain ontologies.
  This was originally done for convenience, however we observe that it may be more clear and effective to only consider the foundational grouping conceptually, and individually import whichever ontologies are required.
- In version 1.2 of the ontology, we replace all imports of the Foundation ontology only with the ontologies that are used. In some cases, this may be equivalent to importing the Foundation ontology, however in other cases this will be a sub-theory of the Foundation ontology.

Land Use Ontology:

- Defined new subclasses of Parcel based on sample data: TrafficZone, PlanningDistrict, Municipality. It's unclear what the logical distinction between these classes will be, but they are distinct types of parcels used by the domain experts.
- Introduced additional land use classifications, aligned with lbcs classifications where possible, based on CLUMP and AAFC systems.

- Added TrafficZone subclass of Parcel
- Added population properties for Parcels

PublicTransit Ontology:

- Added specializations of the Activity class: TransitTrip and TransitIncident
- Included some additional properties and added to the definitions of some existing classes
- Imported Transportation System ontology
- Imported Activity ontology

Time Ontology:

- Revised to reuse the new version of OWL-Time (updated via W3C in fall of 2017)

Parking Ontology:

- Based on requirements identified by CUHK use cases, the parking ontology has been extended to capture additional concepts to provide a more detailed picture of existing car parks.

6 Urban System Characteristics and Behaviour

In the urban system, we recognize the following key concepts that must be defined:

- Person
- Organization
- Household
- Building
- Parking
- Vehicle
- Transportation Networks
- Transit
- Land Use
- Travel

The semantics of each of these concepts will be defined by a generic ontology. These generic ontologies will then be used in the iCity ontology to define the urban system and its behaviour; its population, land use, transportation infrastructure, and the travel that occurs within it. This representation may then be extended to capture the individual iCity applications so that they may be integrated with one another and sufficiently well-defined so as to be shareable and reproducible with the research community. Foundational Ontologies will be also required in order to define the core concepts that apply across the transportation domain. These will be introduced first, followed by the presentation of each generic ontology in more detail. Where warranted, we provide a brief description of the domain and role of the ontology prior to describing its classes and their properties.

To do: include discussion on practices adopted for the reuse of existing ontologies

- Save and re-publish with new IRI to ensure availability and consistency (there may be issues if some ontologies update and do not use Version IRI metdata)
- Entities created for organization: XThing, XObjectProperty, XDataProperty…

6.1 Foundational Ontologies

~~*http://ontology.eil.utoronto.ca/icity/Foundation.owl*~~

In addition to the concepts that are specific to an urban system, there exist foundational concepts that are required to fully define the domain. In particular, the foundational ontology captures the concepts of time, space, change, activities, and resources; each concept is defined its own sub-ontology.

6.1.1 Spatial Location Ontology

http://ontology.eil.utoronto.ca/icity/SpatialLoc.owl

Namespace: geo

To capture generic spatial features, we require 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, we need to be able to describe the spatial relationship between various features (e.g. containment, overlaps). To achieve this, we leverage the GeoSPARQL[3] standard. GeoSPARQL specifies the required vocabulary of spatial relations; it is particularly attractive as it has been published as a standard by the OGC, and in addition it provides specialized functions for querying spatial data according to these relations that are supported by some knowledge base implementations. The Spatial Location Ontology also includes the property hasLocation to capture the relationship between non-spatial objects and the spatial locations they occupy. Similarly, the associatedLocation is introduced to capture the association of some non-spatial object to a particular location. For example, a train station may occupy a fairly large spatial location but be associated with a particular point.

In order to capture the quantitative geospatial information, spatial features may be associated with geometry objects, via the hasGeometry property. These geometries may then be encoded with coordinate information through the specification of WKT (well-known text) values with the data property asWKT. The default reference system for the coordinate values is assumed to be WGS84. In theory, GeoSPARQL supports the identification of alternate reference systems, these are captured as IRIs and concatenated with the coordinates. Currently, knowledge base support for translation between these systems is limited.

Object	Property	Value
geo:Feature	subClassOf	geo:SpatialObject
geo:Geometry	subClassOf	geo:SpatialObject

The ontology specifies properties for the topological relationship between spatial objects, among these are:

Property	Characteristic	Value (if applicable)
geo:sfEquals	Domain and Range	geo:SpatialObject
geo:sfDisjoint	Domain and Range	geo:SpatialObject
geo:sfIntersects	Domain and Range	geo:SpatialObject
geo:sfTouches	Domain and Range	geo:SpatialObject
geo:sfWithin	Domain and Range	geo:SpatialObject
geo:sfContains	Domain and Range	geo:SpatialObject
geo:sfOverlaps	Domain and Range	geo:SpatialObject
geo:sfCrosses	Domain and Range	geo:SpatialObject
geo:hasGeometry	Domain	geo:Feature

[3] http://www.opengeospatial.org/standards/geosparql

	Range	geo:Geometry
hasLocation	Range	geo:Feature
associatedLocation	Range	geo:Feature

RCC8 and Egenhofer relations are also defined.

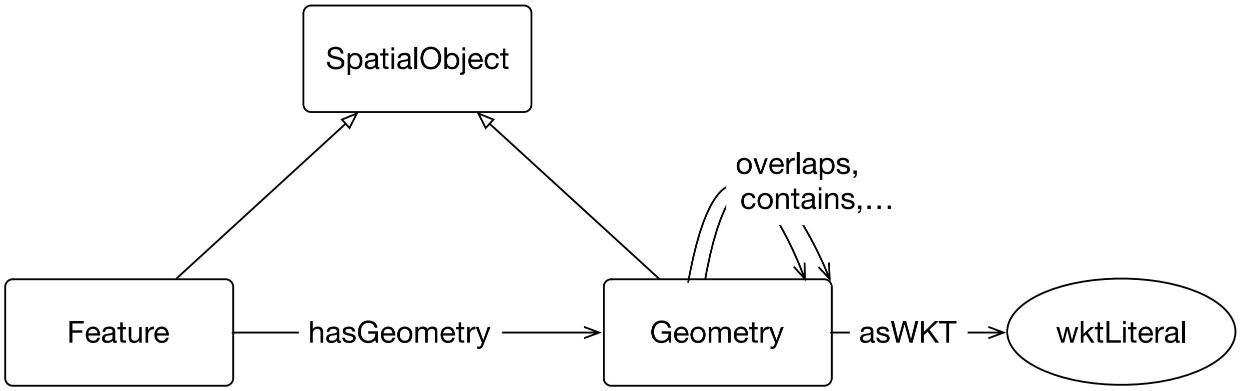


Figure 2: Key classes in geoSPARQL.

TYPE	SHAPE	Geometry Class	SYNTAX
POINT		sf:Point	POINT(longitude latitude)
LINESTRING		sf:LineString	LINESTRING(long1 lat1, long2 lat2, …)
POLYGON		sf:Polygon	POLYGON((long1 lat1, long2 lat2, … , long1 lat1))
POLYGON (WITH HOLE)		sf:Polygon	POLYGON((long1 lat1, long2 lat2, … , long1 lat1), (longA latA, longB latB, …, longA latA))

Figure 3: An example of some basic shape encodings in WKT, from (Kolas & Battle, 2012).

To capture geometries as RDF literals, they are given the `geo:WktLiteral` datatype, for example: "`POINT(-77.03524 38.889468)"^^geo-sf:wktLiteral`.

```
ex:Monument1 a ex:Monument;
   rdfs:label "Washington Monument";
   geo:hasGeometry ex:Point1 .
ex:Point1 a geo:Point;
  geo:asWKT "POINT(-77.03524 38.889468)"^^geo-sf:WktLiteral.
Ex:Park1 a ex:Park;
     rdfs:label "Example Park";
     geo:hasGeometry ex:Polygon1 .
ex:Polygon1 a geo:Polygon;
  geo:asWKT "POLYGON((-77.05 38.87, -77.02 38.87, -77.02
  38.9, -77.05 38.9, 77.05 38.87))"^^geo-sf:WktLiteral.
```

Figure 4: An example encoding of geospatial information for the Washington Monument, from (Kolas & Battle, 2012).

Reused Ontologies:

1. geo: http://www.opengis.net/ont/geosparql#

6.1.2 Time Ontology

http://ontology.eil.utoronto.ca/icity/Time.owl

Namespace: time

- Temporal Entity: A Temporal Entity may refer to an instant or an interval in time.
  A Temporal Entity may be described as being **before** or **after** some other Temporal Entity(s).
  A Temporal Entity has a **beginning** and **ending** time Instant.
  A Temporal Entity has a **duration**.
- Instant
  An Instant may be **inside** some Interval.
- Interval
  An Interval may be described as **before**, **meets, overlaps, starts, during, finishes, equals** some other Interval(s).

Object	Property	Value
time:TemporalEntity	EquivalentClass	time:Instant and time:Interval
	time:before	only time:TemporalEntity
	time:after	only time:TemporalEntity
	time:hasBeginning	only time:Instant
	time:hasEnding	only time:Instant
	time:hasDuration	only time:Duration
time:Instant	subClassOf	time:TemporalEntity
	time:inside	only time:Interval
	time:inTimePosition	max 1 time:TimePosition
	time:inXSDDateTime	max 1 xsd:DateTime
time:Interval	subClassOf	time:TemporalEntity
	time:before	only time:Interval
	time:meets	only time:Interval
	time:overlaps	only time:Interval
	time:starts	only time:Interval
	time:finishes	only time:Interval
	time:during	only time:Interval
	time:equals	only time:Interval
time:DateTimeDescription	time:day	max 1 rdfs:Literal
	time:dayOfWeek	max 1 owl:Thing
	time:dayOfYear	max 1 rdfs:Literal
	time:hour	max 1 rdfs:Literal
	time:minute	max 1 rdfs:Literal
	time:month	max 1 rdfs:Literal
	time:second	max 1 rdfs:Literal
TimePeriod	subClassOf	time:DateTimeDescription
CalendarPeriod	subClassOf	time:DateTimeDescription

Update: Note, version 1.1 updated to reflect revision to owl-time released as a W3C recommendation in October 2017.

Reused Ontologies:

- time: OWL-Time Ontology[4] originally presented by (Hobbs & Pan, 2004)

6.1.3 Change Ontology

http://ontology.eil.utoronto.ca/icity/Change.owl

Namespace: iCity-Change

Many of the concepts identified in the urban system ontologies are subject to change. For example, a Vehicle will have one location at one time, and another location at a later time; it may have only one passenger at one time, and four passengers at a later time. Similarly, many attributes of Persons, Households, and even Transportation Networks are subject to change. An approach to representing changing properties, or *fluents*, that leverages the 4-dimensionalist perspective was proposed by (Welty, Fikes, & Makarios, 2006). We adopt a similar approach, , based on the design pattern presented by (ref WOP) requiring the division of classes that are subject to change into two parts: invariant and variant parts of the concept; we refer to these as TimeVaryingConcept and Manifestation classes, respectively. By distinguishing between these class types and recognizing the properties that are (and aren't) subject to change, the ontology supports the capture of both the static and dynamic aspects of a particular entity.

- TimeVaryingConcept: A class that is subject to change is defined as a type of TimeVaryingConcept (e.g. Vehicle may be a subclass of TimeVaryingConcept). The TimeVaryingConcept itself is invariant and defined by properties that do not change over time. As per (Krieger, 2008), we view TimeVaryingConcepts as perdurants (things that occur over time, i.e. processes).
  A TimeVaryingConcept has **Manifestations** that demonstrate their changing (variant) properties over time. Different types (**subclasses**) of TimeVaryingConcept may be defined based on the Manifestations that are part of them. For example, VehiclePD[5]s have manifestations that are Vehicles.
  A TimeVaryingConcept **exists at** some Interval.
  The class of TimeVaryingConcepts is equivalent to the class of things that have some Manifestations - and *only* Manifestations - in the hasManifestation relation.
- Manifestation: A Manifestation of some TimeVaryingConcept at a particular point/interval in time.
  A Manifestation **exists at** some Instant (or possibly Interval).
  The class of Manifestations is equivalent to the class of things that are manifestations of some TimeVaryingConcept - and *only* time varying concepts - in the manifestationOf relation.

Object	Property	Value
TimeVaryingConcept	disjointWith	time: TemporalEntity and Manifestation
	existsAt	exactly 1 time:Interval
	hasManifestation	only Manifestation
	equivalentClass	hasManifestation some Manifestation and hasManifestation only Manifestation
Manifestation	disjointWith	TimeVaryingConcept and time: TemporalEntity
	equivalentClass	manifestationOf some TimeVaryingConcept and manifestationOf only TimeVaryingConcept
	manifestationOf	only TimeVaryingConcept
	existsAt	exactly 1 time:TemporalEntity

[4] https://www.w3.org/TR/owl-time/

[5] Note: in order to avoid confusion that may result from the use of the "-Process" suffix (e.g. VehicleProcess,OrganizationProcess), we opt instead to use the suffix "PD", i.e. short for "Perdurant".

Property	Characteristic	Value (if applicable)
hasManifestation	inverseOf	manifestationOf
	Inverse Functional	-
manifestationOf	Functional	-
existsAt	Ranges	time:TemporalEntity

Reused Ontologies:

- iCity-Time

6.1.4 Activity Ontology

http://ontology.eil.utoronto.ca/icity/Activity.owl

Namespace: activity

- Activity: An Activity describes something that occurs in the domain.
  An Activity may be further defined by **(decomposed into) Subactivities**.
  An Activity may have **precondition and/or effect** States.
  An Activity may be **enabled by or cause** some States. An enabling of causing state is a generalization of a precondition/effect; an Activity is enabled by or causes some State if it has a subactivity with a precondition or effect (respectively) of that State.
  In other words, the state may not be required directly before, or cause directly after the activity, but by some more specialized sub-activity.
  An Activity **occurs at** some point in time and space.
  An Activity takes place **during some interval**, and so **has** some **duration**.
  An Activity may have some Manifestations that participate in it.
  An Activity may **occur before** some other Activity.
- State: a state refers to a class of manifestations. It may be an immediate precondition or effect of some Activity, or more generally it may enable or be caused by some Activity (in which case, it might be a direct precondition or effect of some subactivity of the activity).
  A state may be complex and refer to some combination of classes of manifestations.
  A note on complex states:
  Say that a shopping activity, Activity-Shop, requires both the state of a vehicle having at least 30L of gas in the tank (let's call this state VehicleW30LGas), but also some state type wherein the mall is open, (we'll call this state OpenMall). Each state type would first be defined separately. This precondition could bet stated as:
    precondition(VehicleW30LGas,Activity-Shop) AND
    precondition(OpenMall,Activity-Shop)
    were the preconditions required disjunctively, we could state:
    precondition(VehicleW30LGas,Activity-Shop) OR
    precondition(OpenMall,Activity-Shop)

  However, in large and complex domains, there will be cases in which the above approach is undesirable. In particular, due to the complexity of the description that results as the state type becomes more detailed. In many cases it will be more natural and convenient to be able to refer to a single, aggregate state. We therefore extend the representation of States to capture aggregation, adopting the following approach used in the description of state trees in TOVE by (Fox, Chionglo, & Fadel, 1993).

A State may be either non-terminal or terminal. A terminal state has no child states, and therefore refers directly to a class of manifestations, whereas a non-terminal state has child states, which may define some classes of manifestations, or further define some other complex state types.

NonTerminalState(x) v TerminalState(x) = State(x)

A state type cannot be both non-terminal and terminal.

TerminalState disjointWith NonTerminalState

- A terminal state has no substates (cannot be decomposed). It corresponds to a particular class of manifestations. A terminal state is achieved at some time if and only if there exists a manifestation within its defined classification, that exists at that time.
- A non-terminal state may be conjunctive or disjunctive. Naturally, a conjunctive state is defined by the conjunction of its child state, whereas a disjunctive state is defined by the disjunction of its child states.

  ConjunctiveStateType(x) v DisjunctiveStateType(x) = NonTerminalStateType(x)

  A state cannot be both conjunctive and disjunctive.

  ConjunctiveStateType disjointWith DisjunctiveStateType

  Conjunctive and disjunctive states, which *do* have substates, are achieved at some time if their decomposition of state is achieved.

  Note that in this representation, *decomp_of* is not a transitive relation, it only refers to the direct children of a non-terminal state. A more general relation that *is* transitive is the *substate* relation.

  decomp_of(x,y) -> substate(x,y)

A note on activity ordering: Some notion of an ordering over activity occurrences must also be captured. To address this, we introduce the properties: "occursBefore" and "occursDirectlyBefore" in the Activity ontology. Although we are unable to completely capture the semantics of this ordering, we can express the following aspects using property chaining:

- An activity occursBefore another if its endOf instant is before the beginOf instant of the other activity; the occursBefore relation is transitive.
  endOf o before o inverse (beginOf) -> occursBefore
- An activity occursDirectlyBefore another if it occursAt an interval that meets the interval of the other activity; occursDirectlyBefore is a subproperty of occursBefore.
  occursAt o intervalMeets o inverse(occursAt) -> occursDirectlyBefore

Object	Property	Value
Activity	hasSubactivity	only Activity
	hasPrecondition	only State
	enabledBy	only State
	hasEffect	only State
	causes	only State
	occursAt	some time:Interval
	beginOf	some time:Instant
	endOf	some time:Instant
	spatial_loc:hasLocation	only spatial_loc:SpatialFeature
	hasParticipant	only change:Manifestation

	occursBefore	only Activity
	occursDirectlyBefore	only Activity
State	preconditionOf	only Activity
	enables	only Activity
	effectOf	only Activity
	causedBy	only Activity
	achievedAt	only time:TemporalEntity
TerminalState	subClassOf	State
	disjointWith	NonTerminalState
	subClassOf	change:Manifestation and (preconditionOf some Activity or effectOf some Activity)
	hasDecomp	exactly 0 StateType
NonTerminalState	subClassOf	State
	disjointWtih	TerminalState
	hasDecomp	only State and min 2 State
	hasSubstate	only State
ConjunctiveState	subClassOf	NonTerminalState
	disjointWith	DisjunctiveState
DisjunctiveState	subClassOf	NonTerminalState
	disjointWith	ConjunctiveState

Property	Characteristic	Value (if applicable)
hasSubactivity	Transitive	-
hasPrecondition	Domains	Activity
	Ranges	State
~~enabledBy~~	~~subPropertyOf~~	~~hasPrecondition~~
hasPrecondition	subPropertyOf	enabledBy
hasEffect	Domains	Activity
	Ranges	State
~~causes~~	~~subPropertyOf~~	~~effectOf~~
hasEffect	subPropertyOf	causedBy
occursAt	Domains	Activity
	Ranges	time:TemporalEntity
hasParticipant	Domains	Activity
	Ranges	change:Manifestation
participatesIn	inverseOf	hasParticipant
preconditionOf	inverseOf	hasPrecondition
enables	inverseOf	enabledBy
effectOf	inverseOf	hasEffect
causedBy	inverseOf	causes
achievedAt	Domains	State
	Ranges	time:TemporalEntity
	superPropertyOf	inverse(preconditionOf) o beginOf

	superPropertyOf	inverse(effectOf) o endOf
	superPropertyOf	inverse (hasDecomp) o conjunctiveAchievedAt o during
hasDecomp	Domains	State
	Ranges	State
	subPropertyOf	hasSubstate
hasSubstate	Domains	State
	Ranges	State
	~~subPropertyOf~~	~~hasDecomp~~
	Transititve	
beginOf	Domains	Activity
	Ranges	time:Instant
	superPropertyOf	occursAt o time:hasBeginning
endOf	Domains	Activity
	Ranges	time:Instant
	superPropertyOf	occursAt o time:hasEnd
terminalAchievedAt	subPropertyOf	achievedAt
	subPropertyOf[6]	existsAt
	Domains	TerminalState
disjunctiveAchievedAt	subPropertyOf	achievedAt
	superPropertyOf	hasDecomp o achievedAt
	Domains	DisjunctiveState
conjunctiveAchievedAt	subPropertyOf	achievedAt
	Domains	ConjunctiveState
occursBefore	range	Activity
	domain	Activity
	transitive	
occursDirectlyBefore	subPropertyOf	occursBefore

Reused Ontologies:

- iCity-Change
- iCity-SpatialLocation

6.1.5 Resource Ontology

http://ontology.eil.utoronto.ca/icity/Resource.owl

Namespace: resource

This ontology provides a generic representation of resources that contain core properties generic across all transportation uses. We take the view presented in the TOVE model (Fadel, Fox, & Gruninger, 1994) that "*...being a resource is not an innate property of an object but a property that is derived from the role the object plays with respect to an activity*". The definition of a resource is dependent on its participation in an activity, so the Resource ontology is in fact an extension of the Activity ontology. In this sense, Resources are a class of manifestations, so that rather than have a specialized Resource-perdurant (PD) class, a Resource is a manifestation of some *other* perdurant class in the ontology. For example, an instance of a Vehicle that is a

[6] equivalent property?

manifestation of some VehiclePD may also be an instance of a resource, whereas some other instance of a Vehicle that is some later manifestation of the same VehiclePD may not be a Resource, or it may be a *different* Resource.

- A Resource is a generic representation of some Thing that can be "used" in an Activity. A Resource may have some Location, amount or availability, according to the definition of the Manifestation or TimeVaryingEntity.
  A Resource must be **classified as** some Resource Type.
  A Resource may **participate in** some Activity.
  A *specific* Resource may be **used in** or **consumed in** some activity. As with the precondition and effect properties defined in the Activity Ontology, these relationships are specific to a particular activity; more general properties may be defined (analogous to enables and causes) should this be required.
  A Resource may have some **associated location**.
  A Resource may have some **owner**.
- A Resource may *either* be a Divisible Resource or a Non-Divisible Resource. On the surface this may seem counterintuitive -- consider a vehicle being used as a non-divisible resource for transportation, and then later as a divisible resource for scrap metal. However, while these examples might refer to the same car over the span of its lifetime, each one in fact refers to a different manifestation of the car, and hence a different resource. The resources differ in their divisibility because each one is defined with respect to a different activity (e.g. travel, versus metal recycling).
  A divisible resource may be used by or consumed by more than one activity, whereas a non-divisible resource may only be used by one activity (i.e. the object may only be used by one activity at a time).
- A Resource Type describes a class of Resources, (intuitively similar to the State Type class).
  A Resource Type may be **usedBy** or **consumedBy** some Activity; the specification of the Resource Type defines the quantity of a particular resource that will be used or consumed by a particular activity.
  If some resource type is used by an activity, then when the activity occurs, there must be some resource of that type that is (partially) not available. If a resource type is consumed by an activity, then the resource and the entity it is a manifestation of (partially) cease to exist by the end of the occurrence.
  usedBy and consumedBy are subproperties of preconditionOf.

Object	Property	Value
Resource	subClassOf	change:Manifestation
	change:existsAt	exactly 1 TemporalEntity
	spatial_loc:hasLocation	only spatial_loc:SpatialFeature
	hasCapacity	only om:CapacitySize
	capacityInUse	only om:CapacitySize
	activity:participatesIn	min 1 activity:Activity
	usedInOccurrence	only activity:Activity
	consumedInOccurrence	only activity:Activity
DivisibleResource	subClassOf	Resource
	disjointWith	NonDivisibleResource

	hasAvailableCapacity	only om:CapacitySize
NonDivisibleResource	subClassOf	Resource
	disjointWith	DivisibleResource
	usedBy	exactly 1 activity:Activity

Property	Characteristic	Value (if applicable)
usedIn	Functional	-
consumedIn	Functional	-
usedBy	subPropertyOf	preconditionOf
consumedBy	subPropertyOf	preconditionOf
hasOwner	domain	Resource
hasAssociatedLocation	domain	Resource

Reused Ontologies:

- iCity-Activity

6.1.6 Mereology Ontology

http://ontology.eil.utoronto.ca/icity/Mereology.owl

Namespace: mereology

Notions of parthood are ubiquitous in and beyond the transportation domain. While sometimes conflated, there are clear distinctions which can be made between different types of parthood. The mereology ontology focuses on identifying these differences and making them explicit. In this ontology, we identify the following different types of parthood: proper-part-of, component-of, and contained-in. The distinction between these types of parthood may be best explained with the use of examples. An item may be *contained in* my car, but that does not make it a *component of* my car. For example, we may wish to describe passengers or cargo being *contained in* a vehicle, but this relation must be distinguished from the parts and components that make up a vehicle. Similarly, the front of my car is intuitively a part of my car, but not a component of my car. While we may define components of a vehicle, different zone systems (wards, postal codes) are not components, but proper parts of larger areas.

A more detailed analysis reveals clear, ontological distinctions between each of these relations that may formalized clearly with a set of first-order logic axioms. This analysis, presented in [ref Bittner & Donnely] also identifies the expressive limitations of OWL, which prevent a complete representation of this semantics, and discussed the various possible approximations. It is important to consider what should be captured, and what distinctions should be made in the introduction of properties, in contrast with what is actually expressible in the logic. Since we cannot completely capture the required semantics in OWL, some trade-off(s) is required for any partial specification, (e.g. OWL only allows the specification of transitivity for simple object properties).

The difficulty with such an approximation is that the resulting theory defines a semantics for something else entirely. Inherently, some semantics are omitted, which may not be required for one application but may be important for another. For example, if transitivity is a key aspect of some required reasoning, then perhaps a parthood relation would be defined as transitive, and some omissions would be made with respect to the formalization other restrictions (e.g.

cardinality) that should be applied to the parthood relation. Certainly, the use of approximations will be required in some cases, for example in order to support some desired reasoning problems. However, precisely which axiomatization is most suitable will vary between different usage scenarios. We therefore omit a detailed, partial axiomatization in favour of an under-axiomatized specification of the key relations, in order to avoid prescribing one trade-off over another. This leaves the commitment open-ended and variable to suit individual applications' needs.

This ontology defines the general properties such that the commonality between domain-specific part-of relations may be captured, and more detailed semantics may be defined in extensions of the properties. This creates a means of indicating the intended semantics of a relation by identifying the *type* of parthood that it is intended to capture, while allowing for the specification of different partial approximations of the semantics (and possibly also specializations of this semantics), as required. For example, a notion of parthood arises in the description of a building and the units it is divided into. In this case, this relationship may be identified as a sort of hasComponent relation; a new property 'hasBuildingUnit' may be identified then as a subPropertyOf hasComponent. We are free to assess, for the 'hasBuildingUnit' relation, which approximations of the component-of relation are the most suitable. The approximation chosen for one type of parthood relation does not constrain the choice of approximation for another.

In addition to the aforementioned work by [ref Bittner & Donnelly], various approaches to the partial capture of these mereological relationships in OWL have been proposed that may be used to extend the ontology presented here, such as in [https://www.w3.org/2001/sw/BestPractices/OEP/SimplePartWhole/], and also in top-level ontologies such as [ref BFO].

Property	Characteristic	Value (if applicable)
properPartOf	inverseOf	hasProperPart
hasProperPart	inverseOf	properPartOf
componentOf	subPropertyOf	properPartOf
	inverseOf	hasComponent
hasComponent	subPropertyOf	hasProperPart
	inverseOf	componentOf
immediateComponentOf	subPropertyOf	componentOf
containedIn	inverseOf	contains
contains	inverseOf	containedIn
immediatelyContainedIn	subPropertyOf	containedIn

Future work:

- Extend to capture multidimensional mereotopological relations as in CODI
    - Map the relations to GeoSPARQL properties
- Extend with a first-order formalization or mereology

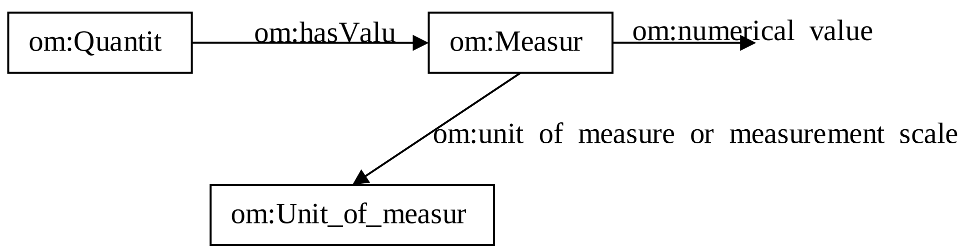
6.1.7 Ontology of Units of Measure

http://ontology.eil.utoronto.ca/icity/OM.owl

Namespace: om

The Ontology of Units of Measure provides a structured vocabulary to describe, among other things, the different values (measures) that we associate to given quantities. This allows us to provide greater detail regarding specific measurements that are defined in the ontology. Rather than simply have a simple data property to describe the length of some road segment as "10 m", with the units of measure ontology we are able to describe the nature of the quantity (i.e. length), its value as a Measure (10 m), and also describe the unit that the measure's numerical value is given in (e.g. metres).
In the context of the iCity ontology, quantities, units, and/or measures that are defined with domain-specific concepts (e.g. vehicles, lanes) are defined by reusing and extending the units of measure ontology in the relevant modules.

- A quantity has some measured value
- A measured value (om:Measure) is associated with a unit of measure
- There are many types (subclasses) of units of measure, such as length, mass, speed, and currency.



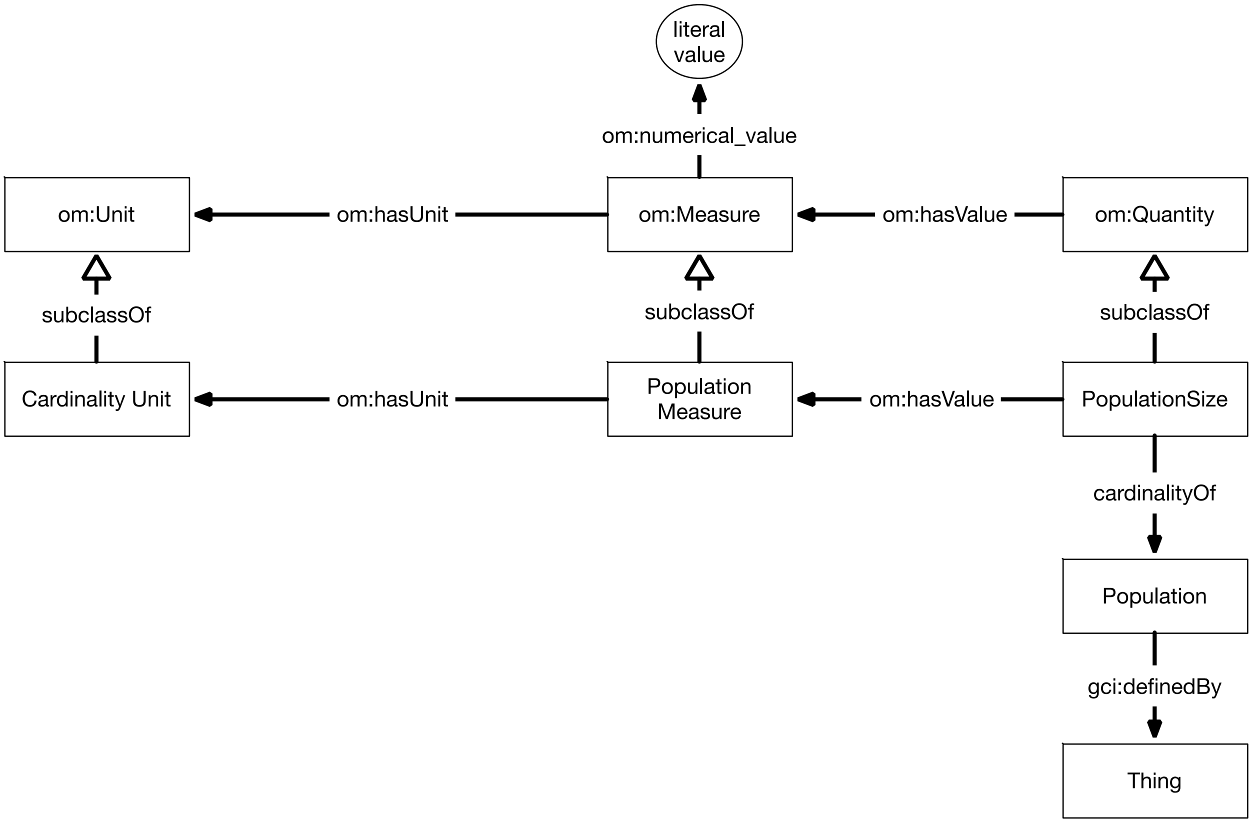


Figure 5: Representation of populations as reused from the GCI Ontology.

In order to represent populations, we reuse the following classes from the GCI-Foundation ontology: ~~govstat:Population,~~ gci:PopulationSize, gci:PopulationSizeMeasure, and gci:CardinalityUnit. Refer to the working paper on the GCI Ontology for more details on this approach. The meaning of population is general here, while it may define a population of residents within some zone, it may also be used to describe the population of vehicles occupying some stretch of the road network.

The quantity of interest (population size being measured/described) is defined as gci:Population_Size, a subclass of Quantity. Population_Size has some unit of measure (a cardinality unit), and has_value some Population_Measure (with an associated numeric value). The elements associated with a population quantity are captured through the defined_by property that relates a Population to some class of objects. For example, consider the measurement of the number of cars on some road segment, we could specify: Population_Size and cardinalityOf only (Population and definedBy only (Vehicle)). The defining population might be even more precisely captured for a given Road Segment, X, as depicted in Figure 6: definedBy only (Vehicle and onSegment value X). These specializations are defined, as required, within the relevant module; for example, a vehicle population would be defined in a module that contains the required concepts of vehicles and road segments. The units of measure ontology captures only to core concepts of Population Size, Population Measure, Cardinality Unit, and Population, as depicted in Figure 5.

Capacity and its associated quantity and measure are defined similar to population.

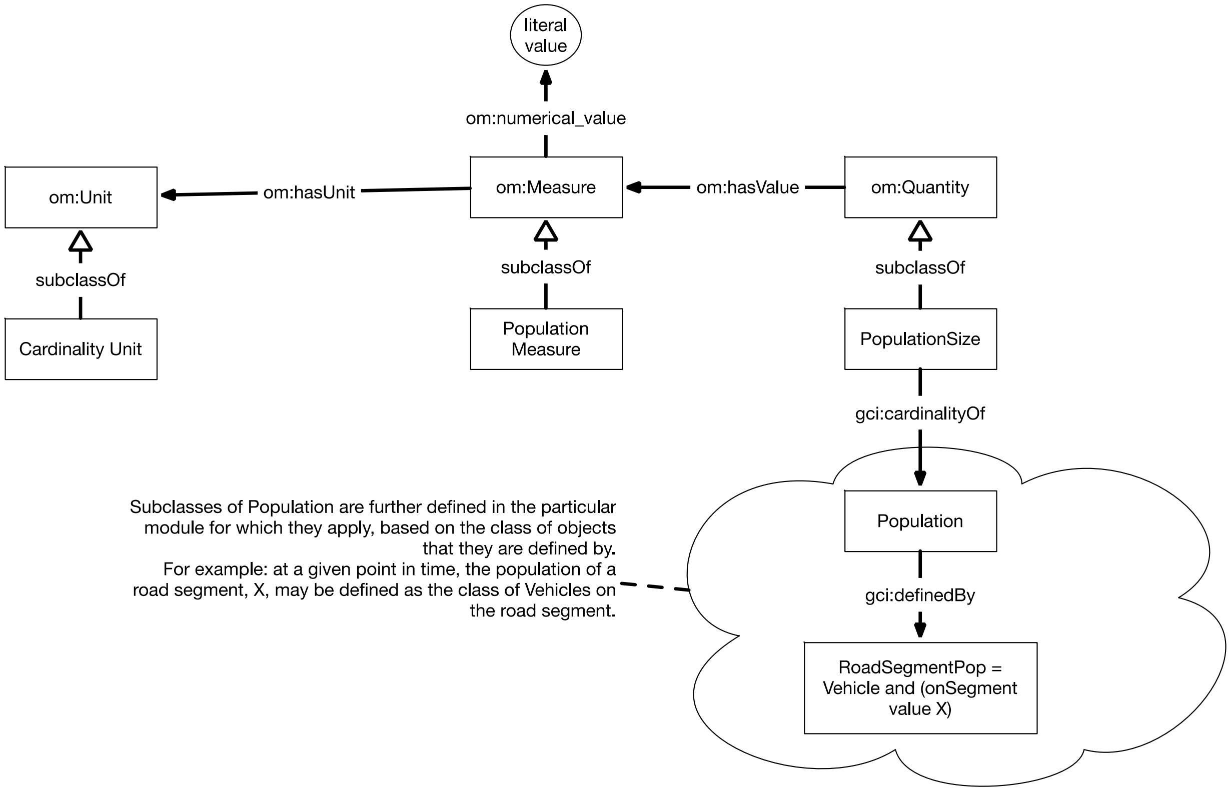


Figure 6: Specialization of populations.

Revisions:

- Updated to OM2.0
- OM:MonetaryValue a om:Unit_of_measure
- Monetary Value: Monetary Values may be attributed to things such as the purchase of a dwelling, or the salary of some Job.
  A Monetary Value has a **dollar value** relative to a particular **date** (year).
  A Monetary Value has some associated **currency**.
- Population (via GCI): The population class is defined as a subclass of om:Quantity. It represents the quantities of counted sets. Various subclasses of Population will be defined as required, e.g. Resident_Population, Vehicle_Population, etc. A population has a measure (Population_measure) in some cardinality unit
- Population_measure
- Population_unit_of_measure
- Cardinality_unit
- Capacity: subclass of Quantity and hasUnit CardinalityUnit
- CapacityRate, e.g.
  - OM:lanecap a om:Unit_of_measure (vehicles/hour)
- Added quanitites: RoadOccupancy, VehicleVolume, MeanTravelSpeed
- Added aggregateOver and aggregateOf properties in order to defined aggregate quantities in greater detail

Future work:

Consider whether it is more accurate to describe the position coordinates as quantities that are measured in degrees *that are relative to a geodetic datum* (e.g. NAD83). In this approach, we are considering "degrees relative to a particular datum" as a kind of measure. In any case, it is important that we are able to distinguish between different position systems as latitudinal and longitudinal values cannot be assumed to use the same sort of system. In particular, WGS84 and NAD83, which were originally nearly equal are now considerably different (depending on the area) due to changes that have occurred to the earth since 1984. Note that http://data.ign.fr/def/ignf/20150505.en.htm may be a relevant ontology.

Object	Property	Value
om-2:Quantity	om-2:hasValue	only om-2:Measure
om-2:Measure	om-2:hasUnit	only om-2:Unit
om-2:Length_unit	subClassOf	om-2:Unit
om-2:Mass_unit	subClassOf	om-2:Unit
om-2:Area_unit	subClassOf	om-2:Unit
om-2:Acceleration_unit	subClassOf	om-2:Unit
om-2:Volume_unit	subClassOf	om-2:Unit
om-2:Speed_unit	subClassOf	om-2:Unit
om-2:Amount_of_money_unit	subClassOf	om-2:Unit
Geo_Position_unit	subClassOf	om-2:Unit
gci:Cardinality_unit	subClassOf	om-2:Unit
om-2:UnitDivision	subClassOf	om-2:Unit
Cardinality_unit_per_time	subClassOf	om-2:UnitDivision
	om-2:hasNumerator	only gci:Cardinality_unit
	om-2:hasDenominator	only om-2:TimeUnit
...	subClassOf	om-2:Unit_of_measure
MonetaryValue	subClassOf	om-2:Measure
	hasRelativeYear	exactly 1 xsd:gYear
	om-2:hasUnit	only om-2:Amount_of_money_unit
gci:Population_measure	subClassOf	om-2:Measure
	subClassOf	CardinalityMeasure
CardinalityMeasure	subClassOf	om-2:Measure
	hasUnit	only gci:Cardinality_unit
ValueOfMoney	subClassOf	om-2:Quantity
	subClassOf	om-2:AmountOfMoney
	om-2:hasValue	only MonetaryValue
om-2:Length	subClassOf	om-2:Quantity
	om-2:hasValue	only (om-2:Measure and om-2:hasUnit only om-2:Length_unit)
gci:PopulationSize	subClassOf	om-2:Quantity
	om-2:hasValue	only gci:Population_measure

	gci:cardinalityOf	exactly 1 gci:Population
CapacitySize	subClassOf	om-2:Quantity
	om-2:hasValue	only gci:Cardinality_measure
	gci:cardinalityOf	exactly 1 Capacity
CapacityRate	subclassOf	om-2:Quantity
	om-2:hasValue	only (om-2:hasUnit only CardinalityUnitPerTime)
	gci:cardinality_of	exactly 1 Capacity
om-2:Mass	subClassOf	om-2:Quantity
	om-2:hasValue	only (om-2:hasUnit only om-2:Mass_unit)
om-2:Area	subClassOf	om-2:Quantity
	om-2:hasValue	only (om-2:hasUnit only om-2:Area_unit)
om-2:Volume	subClassOf	om-2:Quantity
	om-2:hasValue	only (om-2:hasUnit only om-2:Volume_unit)
om-2:Acceleration	subClassOf	om-2:Quantity
	om-2:hasValue	only (om-2:hasUnit only om-2:Acceleration_unit)
om-2:Speed	subClassOf	om-2:Quantity
	om-2:hasValue	only (om-2:hasUnit only om-2:Speed_unit)

Property	Characteristic	Value (if applicable)
om-2:hasBaseUnit	domain	om-2:System_of_units
om-2:hasBaseUnit	range	om-2:Unit
om-2:hasDenominator	domain	om-2:UnitDivision
om-2:hasDenominator	range	om-2:Unit
om-2:hasNumerator	domain	om-2:UnitDivision
om-2:hasNumerator	domain	om-2:Unit
om-2:hasAggregateFunction	domain	om-2:Quantity
	range	om-2:function
aggregateOf	domain	om-2:Quantity
aggregateOver	domain	om-2:Quantity

Reused Ontologies:

- ~~om: Ontology of Units of Measure~~[7]
- om-2: Ontology of Units of Measure 2.0[8]

[7] om:http://www.wurvoc.org/vocabularies/om-1.6/
[8] om: http://www.ontology-of-units-of-measure.org/resource/om-2/

- gci: Global City Indicators Foundation Ontology[9]

~~6.1.8 Monetary Value Ontology~~

~~*http://ontology.eil.utoronto.ca/icity/MonetaryValue.owl*~~

~~**Namespace: monetary**~~

- Removed. Monetary Value now integrated with Units of Measure Ontology

6.1.9 Recurring Event ontology

A specification of recurring events, in particular those that are defined according to calendar dates (e.g. every Monday, every March), is required in order to capture information regarding hours of operation, road restrictions, restrictions on parking policies, etc. The design of this ontology was inspired by previous work on an ontology for city services[10] for the Global City Indicator Ontology [ref]. The city-services definition of ServiceDeliveryEvent incorporates and references the representation of recurring events from http://www.vertabelo.com/blog/technical-articles/again-and-again-managing-recurring-events-in-a-data-model, however there are issues in the formalization that prevent its direct reuse:

- Specific to service events: we need a generic representation of repeating, calendar events. The notion of exceptions is specific to a kind of event (such as service delivery).
- Use of Date-time interval: owltime:DateTimeInterval defines a specific interval in time, therefore a recurring event occurs at multiple DateTimeIntervals, not one. For example, there is the interval 08:00-14:00 on August 8, 2018, and then there is *another* interval 08:-14:00 on August 9, 2018.
- Monthly events are underspecified. A monthly event repeats on a given day of the month (e.g. the 1st day of each month). There is the related notion of a repeating event that occurs with respect to some day of the week, e.g. an event occurs on the second Tuesday of each month, however this is distinct from the monthly recurring events defined here. This type of recurrence requires a specialized approach to address (e.g. accounting for week numbers) and is out of the scope of the current representation. The related concept of a “separation count” (as described in the referenced data model) that is required to define repetition such as biweekly or bi-monthly occurrences is also not currently required or addressed here.
- Missing the notion of a yearly event: a yearly event occurs on a specific day of some month.

A specification of recurring events, in particular those that are defined according to calendar dates (e.g. every Monday, every March), is required in order to capture information regarding hours of operation, road restrictions, restrictions on parking policies, and so on. The design of this ontology was inspired by previous work on an ontology for city services[11] for the Global City Indicator (GCI) Ontology [ref], however due to incompatibilities in the scope and semantics of the GCI ontology we do not directly reuse it in the TPSO. The GCI Ontology defines recurring events specifically as “Service” events, whereas the transportation requires a more general notion of recurring events. The GCI Ontology employs the concept of a time interval to

[9] http://ontology.eil.utoronto.ca/GCI/Foundation/GCI-Foundation-v2.owl#

[10] http://ontology.eil.utoronto.ca/city-services/city-services.owl#

[11] http://ontology.eil.utoronto.ca/city-services/city-services.owl#

capture when some event recurs, however we observe that this is misleading as recurring events will occur at *multiple* intervals in time. In the TPSO, we opt for a more precise representation that identifies the individual occurrences (that occur at a particular time interval) of some recurring event.

The Recurring Event Ontology adopts the following representation of recurring events: daily, weekly, and monthly recurring events (and their related properties) are defined, however the ontology may be extended with similar definitions of other type of recurring events, as required. This approach is based on the GCI Ontology work and adapted to provide a more suitable and complete representation of recurring events for the transportation domain.

An instance of a recurring event corresponds to a class of activities (e.g., all of the occurrences of a Tuesday, all of the occurrences of the weekly waste pickup). The intuition is that the occurrences of a recurring event are all the same type of activity. What defines a recurring event is a combination of the activity type (e.g. a transit trip from point A to point B or the provision of a service) and the frequency at which it recurs.

The ontology captures the associated activity type with the *hasOccurrence* property that relates recurring events to activities. Classes of recurring events may be captured by identifying their associated classes of Activities, while individual recurring events may be associated with one or more instances of an activity.

The Recurring Event ontology reuses the Activity ontology, as the concept of an activity is central to the notion of a recurring event: the activities are the recurrences. It is important to note that while the concept of Activity defined in the Activity ontology and is necessary for the definition of a RecurringEvent, it is *not* the case that the concept of RecurringEvents is required for the definition of an Activity. This allows the TPSO to maintain a simpler representation of events in cases where the notion of recurrence not be required.

Recurring events are also identified based on the regular interval at which they occur; this is captured using some combination of the hasTime, dayOfWeek, hasMonth, and dayOfMonth properties. Using these properties, ontology supports definitions of specializations of the RecurringEvent class. In particular, subclasses for daily, weekly, monthly, and yearly recurring events are defined; other classes of recurring events may be defined similarly, as required.

- A DailyRecurringEvent occurs every day. It has a maximum of one associated time – the start time. Typically, a daily recurring event will occur at the same time every day, however there may be no commitment to a recurring start time for the event, in which case no start time is specified. A DailyEvent does not necessarily have a recurring end time (this would require a constant duration), therefore this is not part of the definition (although it is possible to specify).
- A WeeklyRecurringEvent recurs regularly on the same day of the week, as specified by the schema:dayOfWeek property.
- A MonthlyRecurringEvent recurs regularly on the same day of each month, as specified by the dayOfMonth data property. Note that there is often ambiguity regarding the semantics of a monthly recurring event: in this formalization, a MonthlyRecurringEvent is any event that recurs regularly on the same *day* of each month; other interpretations sometimes consider events that recur on the same day of week, or first or last day, in which case the day of month will vary. Such a representation is not included in this ontology, but could be captured in an extension.
- A YearlyRecurringEvent recurs regularly on the same day of the same month, as specified by the hasMonth and dayOfMonth properties. As with MonthlyRecurringEvent,

there may be ambiguity regarding the semantics of a yearly recurring event, however this formalization captures only the notion of an event that recurs on the same day of the same month (e.g. a birthday).

Exceptions to recurring events may also be defined. For example, a business may normally operate on Monday-Friday, except for public holidays. Exceptions may also be defined on *specific* dates (e.g. June 23, 2018), for example due to construction. If applicable, exceptions may be defined for recurring events with the recursExcept property. Conversely, so-called "exceptions" may involve an additional, unusual occurrences. This is captured with the recursAddition property.

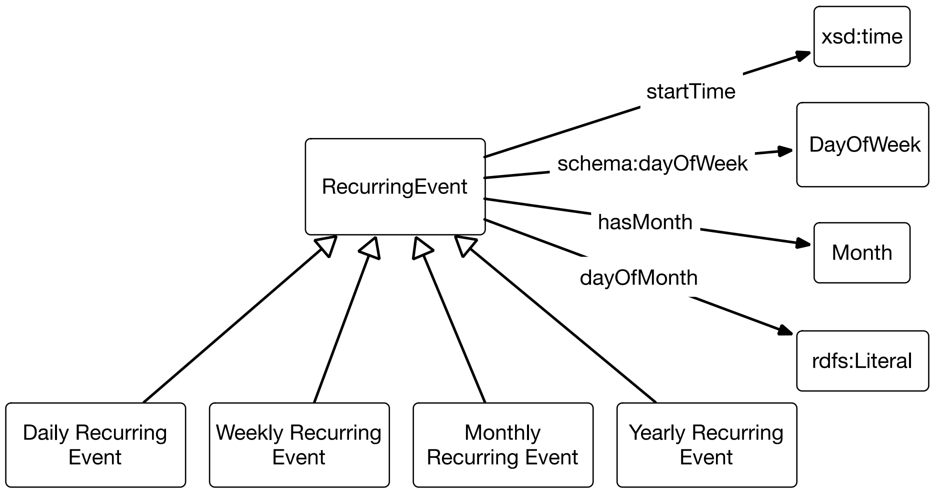
As with an Activity, a RecurringEvent may be decomposed/decomposed into simpler/more complex RecurringEvents to support varying levels of granularity. This decomposition may be specified with the hasSubRecurringEvent property.

Although intuitively similar, it should be noted that calendar concepts are distinct from recurring events. As discussed in earlier work by [Gruninger et al. KR paper on dates & durations], the semantics of calendar dates – concepts such as "Monday", or "January" – can be interpreted using a theory of activities. For example, Monday may be described as an event that recurs every 7 days, and has a duration of 24 hours.

Finally, it is important to note that RecurringEvents such as scheduled transit trips and operating hours represent planned or usual occurrences. For example, while a business may be open at some recurring intervals, it's possible that given some exceptional circumstances (e.g power failure) they may not be open during the predefined days and times.

Object	Property	Value
RecurringEvent	hasOccurrence	only activity:Activity
	spatial_loc:associatedLocation	only spatial_loc:Feature
	hasSubRecurringEvent	only rec:RecurringEvent
	startTime	only xsd:time
	endTime	only xsd:time
	schema:dayOfWeek	only DayOfWeek
	endDayOfWeek	only DayOfWeek
	hasMonth	only Month
	endMonth	only Month
	dayOfMonth	only rdfs:Literal
	endDayOfMonth	only rdfs:Literal
	beginsRecurring	only time:TemporalEntity
	endsRecurring	only time:TemporalEntity
	recursExcept	only time:TemporalEntity or DayOfWeek
	recursAddition	only time:TemporalEntity or DayOfWeek
DailyRecurringEvent	subclassOf	RecurringEvent
	startTime	max 1 xsd:time
WeeklyRecurringEvent	subclassOf	RecurringEvent
	schema:dayOfWeek	exactly 1 DayOfWeek

MonthlyRecurringEvent	subclassOf	RecurringEvent
	dayOfMonth	exactly 1 rdfs:Literal
YearlyRecurringEvent	subclassOf	RecurringEvent
	hasMonth	exactly 1 Month
	dayOfMonth	exactly 1 rdfs:Literal



Future work:

- Address the relationship between a Recurring (Service) Event and an Event (Activity) in more detail. Based on the properties of a recurring event, additional constraints on its occurrences (related activities) may be inferred.
- Additional temporal constraints may be specified to describe the relationship between a Recurring Event and its sub-Recurring Events: the sub-Recurring Events may only recur during the times at which the Recurring Event recurs.
- An ordering relationship over sub-Recurring Events may be useful in future implementations, however this is not currently captured or required.

6.1.10 Observations Ontology

http://ontology.eil.utoronto.ca/icity/Observations

Namespace: obs

In the TPSO, the Observations ontology currently only plays a role in the Transportation System ontology to define the sensors in the transportation network. Nevertheless, the Observations ontology is included with the Foundational Ontologies. This is due to the importance of data collection for transportation planning activities. Data collection efforts take various forms – whether through surveys, manual canvassing, or the use of sensors. With a growing access to the Internet of Things, data from available sensors will continue to expand, likely to include observations about persons, vehicles, and so on. It is important to not only capture the collected data, but the source of the observations.

The Observations ontology reuses the SSN (Semantic Sensor Network) ontology directly from http://www.w3.org/ns/ssn/. SOSA is the foundation for the SSN (Semantic Sensor Network) and provides the scope required for this application, however it does not specify any definitions for the terms, therefore we have opted to reuse the SSN Ontology. The SSN Ontology is a W3C recommendation that has been widely adopted to represent sensors and their observations. It is this widespread use which has motivated the adoption of the SSN ontology to capture sensors and their observations in the domain of transportation planning. The relevant terms are highlighted here.

- Sensor: A Sensor is a device that makes some observation and may be triggered by some stimulus.
- Observation: An Observation has some feature of interest – the thing whose property is being detected by the sensor. An observation observes some ObservableProperty. A phenomenon time (i.e. the time at which the property was demonstrated) and result time may be associated with an observation.

The Observations Ontology generalizes concepts from the SSN Ontology in order to expand the representation to include observations collected without the use of a sensor. In doing so, the concept of Observer is introduced; Observer is a generalization of a Sensor and could also include concepts such as Persons or Surveys.

Note that the SSN ontology does not make any commitments as to whether instances of ssn:Property should be generic (e.g. ex:temperature) or specific to the feature of interest (e.g. ex:mybodytemperature). Current documentation suggests that this is a choice for the modeler. We opt to define instances of ssn:Property at a general level; this will enable the querying of sensors that observe some property (e.g. vehicle presence) regardless of the location. This is useful as there may be different kinds of sensors that observe the same properties (e.g. loop detectors vs Bluetooth sensors) and while they might not share the exact feature of interest, they may be in close enough proximity to be related and so a property indicating their similarity is desirable.

We generalize the SSN ontology as the scope of observations goes beyond properties that are detected by a sensor. To do this, we introduce a general Observation class, of which sosa:Observation is a subclass; a general Observer class, of which sosa:Sensor is a subclass; and a general observedBy property, or which sosa:madeBySensor is a property. An Observation may be made by a Sensor, however it may also be made by a Person. We anticipate that the representation of these general observations will expand as we investigate different use cases.

Object	Property	Value
Observation	observedBy	only Observer
Observer	inverse(observedBy)	only Observation
sosa:Sensor	subclassOf	Observer
	sosa:madeObservation	only sosa:Observation
	sosa:observes	only sosa:ObservableProperty
	ssn:detects	only ssn:Stimulus
sosa:Observation	subclassOf	Observation
	sosa:madeBySensor	exactly 1 sosa:Sensor
	sosa:hasFeatureOfInterest	exactly 1 owl:Thing and only sosa:FeatureOfInterest
	sosa:hasResult	exactly 1 owl:Thing and only sosa:Result

	sosa:observedProperty	exactly 1 owl:Thing and only sosa:ObservableProperty
	sosa:phenomenonTime	exactly 1 owl:Thing
	sosa:resultTime	exactly 1 rdfs:Literal
	ssn:wasOriginatedBy	exactly 1 owl:Thing and only ssn:Stimulus
sosa:ObservableProperty	subClassOf	ssn:Property
	inverse ('is proxy for')	only ssn:Stimulus
	inverse ('observed property')	only sosa:Observation
	sosa:'is observed by'	only sosa:Sensor
sosa:FeatureOfInterest	ssn:'has property'	min 1 owl:Thing and ssn:Property
sosa:Result	sosa:'is result of'	min 1 owl:Thing

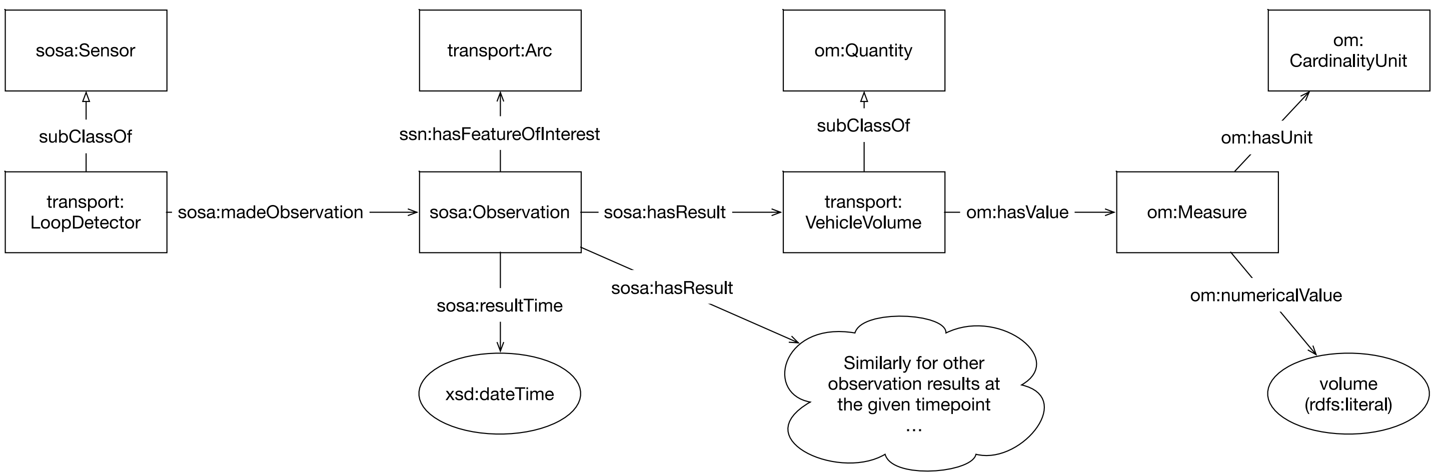


Figure 7: Example of an extension of the Sensor Ontology to capture a sensor in the Transportation System Ontology.

Reused Ontologies:

- SSN
- SOSA

Future work:

Add logic to relate the values of observable property, observation, feature of interest, and result: the observable property indicates how (by what property) the result relates to the feature of interest; e.g. the location of the loop detector indicates the identity of the feature of interest of its observations.

6.2 Contact Ontology

http://ontology.eil.utoronto.ca/icity/Contact

Namespace: contact

Contact information is relevant for a range of concepts in the transportation domain. For example, a building may have some associated address, similarly a person or an organization may have some contact address (or phone number, email, etc). Note that a person's contact address may differ from their place of residence. The iContact ontology is reused to provide the core concepts necessary to define this type of information. The Contact ontology uses concepts

from the spatial location ontology in order to associate an address with a location. It also introduces a more specific definition of hours of operation as a specialization of the RecurringEvent class.
To do: import and extend as required for Person, Building, Organization(?) and Parking ontologies

Object	Property	Value
contact:Address	hasStreetNumber	exactly 1 xsd:nonNegativeInteger
	hasStreet	only xsd:string
	hasCity	exactly 1 schema:city
	…	
	spatialloc:hasLocation	exactly 1 geo:Feature
	subClassOf	iContact:Address
contact:HoursOfOperation	subClassOf	icontact:HoursOfOperation
	subClassOf	rec:RecurringEvent

Reused Ontologies:

- iContact: http://ontology.eil.utoronto.ca/icontact.owl
- iCity Spatial Location: *http://ontology.eil.utoronto.ca/icity/SpatialLoc/*

Future Work:

In future extensions it may be useful to consider the addition of properties such as the time zone (time:TimeZone) associated with an address, as well as the primary language of correspondence.

6.3 Person Ontology

http://ontology.eil.utoronto.ca/icity/Person

Namespace: person

- Person: A Person may have a **unique identifier**.
  A Person has a **date of birth**, and may have a **date of death**.
  A Person has a **mother** and **father**, and may have a **spouse** and/or **child**(ren). Note that we define the parent relation as the legal relation as opposed to biological. This property may be specialized and restricted, for example hasBiologicalMother: exactly 1 Person.
  A Person may **have** some **Job** and associated **Income**.
  A Person has an **address** of residence and may have other contact information such as **E-mail**, **phone number**, etcetera.
  A Person has some age and exactly 1 sex, and sex may be one of only male or female. The definition of sex is distinct from that of a person's gender: "Sex refers to sex assigned at birth. Sex is typically assigned based on a person's reproductive system and other physical characteristics."[12] Future extensions may incorporate a representation of gender, should it be required.
  A person has some Age may or may not be a licensed driver.

Object	Property	Value
PersonPD	subclassOf	change:TimeVaryingConcept
	equivalentClass	change:hasManifestation some Person and change:hasManifestation only Person
	change:existsAt	exactly 1 time:Interval
	hasPersonID	only PersonId
	schema:birthDate	exactly 1 time:Instant
	hasSex	exactly 1 Sex

[12] http://www23.statcan.gc.ca/imdb/p3Var.pl?Function=DEC&Id=24101

	schema:deathDate	max 1 time:Instant
Person	equivalentClass	change:manifestationOf some PersonPD and change:manifestationOf only PersonPD
	subclassOf	change:Manifestation
	change:existsAt	exactly 1 time:TemporalEntity
	hasAge	exactly 1 om:duration
	isLicensedDriver	exactly 1 xsd:boolean
	schema:parent	only Person
	schema:spouse	only Person
	schema:children	only Person
	hasIncome	only MonetaryValue
	schema:address	some schema:PostalAddress
	hasSkill	only Skill
	hasQualification	only Qualification
Sex	equivalentClass	{person:male, person:female}

Reused Ontologies:

- schema.org[13] (A vocabulary as opposed to an ontology)
- Change ontology
- Units of measure ontology
- Time ontology

Future work:

- Attributes such as isLicensedDriver are currently captured as (Boolean) data properties. Future extensions may capture these attributes as object properties, should a more detailed representation be required (e.g. the introduction of a DriversLicense class, with attributes such as its category, expiration date, province of issue, etc). This possibility for future extension applies to many of the defined data properties in the icity ontologies in general.

6.4 Household Ontology

http://ontology.eil.utoronto.ca/icity/Household.owl

Namespace: household

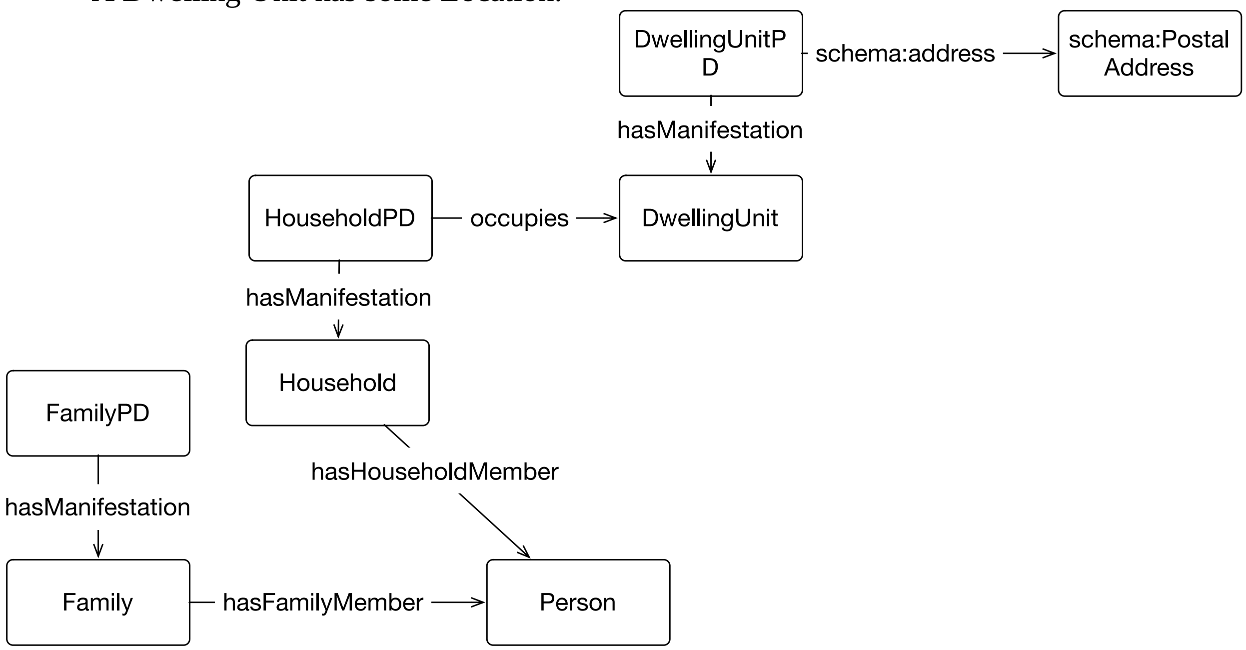
In order to define a Household, we require the following classes and properties:

- Family: The notion of Family simply makes the commitment that it is a group of people who are connected via the has-spouse or has-child properties. From these, we can derive grandparents, aunts, uncles, etcetera.
  One question to consider is to what degree the general/extended Family concept makes sense or is useful. After a few generations the concept of a family will become quite large and confusing, with Persons belonging to many different Families. It may be more useful to consider a relatedTo property between Persons, or only defining restricted subclasses of Family; for example, different types of Family (e.g. Immediate, Extended) may be defined.
- Household: A Household **occupies** a particular Dwelling, according to some **tenure** type. It is defined by this location, so that if the members move (even collectively), the new residence constitutes a new Household.

13 http://schema.org/

Note that a Household, and likely many other classes may have different definitions in different contexts/applications. To address this we may be required to introduce specializations of the class (e.g. ILUTE_Household, TTS_Household) in future extensions.

- Dwelling Unit: A Dwelling Unit is **occupied** by a Household.
  A Dwelling Unit has a **market value**.
  A Dwelling Unit has some Location.



Object	Property	Value
FamilyPD	subclassOf	change:TimeVaryingConcept
	equivalentClass	change:hasManifestation some Family and change:hasManifestation only Family
	change:existsAt	exactly 1 time:Interval
Family	subclassOf	change:Manifestation
	equivalentClass	change:manifestationOf some FamilyPD and change:manifestationOf only FamilyPD
	change:existsAt	exactly 1 time:TemporalEntity
	hasFamilyMember	min 2 person:Person
HouseholdPD	subclassOf	change:timeVaryingConcept
	equivalentClass	change:hasManifestation some Household and change:hasManifestation only Household
	change:existsAt	exactly 1 time:Interval
	occupies	exactly 1 DwellingUnit
Household	subclassOf	change:Manifestation
	subClassOf	gci:Household
	equivalentClass	change:manifestationOf some HouseholdPD and change:manifestationOf only HouseholdPD
	change:existsAt	exactly 1 time:TemporalEntity

	hasHouseholdMember	only person:Person and some person:Person
DwellingUnitPD	subclassOf	change:TimeVaryingConcept
	subclassOf	building:BuildingUnitPD
	equivalentClass	change:hasManifestation some DwellingUnit and change:hasManifestation only DwellingUnit
	change:existsAt	exactly 1 time:Interval
	schema:address	only schema:PostalAddress
	spatial_loc:hasLocation	only spatial_loc:SpatialFeature
DwellingUnit	subclassOf	change:Manifestation
	subclassOf	building:Building and building:BuildingUnit
	equivalentClass	change:manifestationOf some DwellingUnitPD and change:manifestationOf only DwellingUnitPD
	change:existsAt	exactly 1 time:TemporalEntity
	occupiedBy	exactly 1 Household
	hasValue	only monetary:MonetaryValue
	tenureType	only Tenure

Property	Characteristic	Value (if applicable)
occupiedBy	inverseOf	Occupies
hasFamilyMember	subPropertyOf	mer:hasComponent
hasHouseholdMember	subPropertyOf	mer:hasComponent

Future Work:

- Extend the definitions of the classes beyond OWL to capture the different notions of family membership and the types (subclasses) of Family that result.
    - hasParent subpropertyOf isRelated
    - hasChild subpropertyOf isRelated
    - hasParent o hasChild subpropertyOf isRelated
    - hasChild o hasParent subPropertyOf isRelated
    - hasParent o (hasParent)- subPropertyOf isRelated
    - …

Reused Ontologies:

- schema.org
- gci: GCI-Shelter Ontology[14]
- mer:Mereology Ontology

6.5 Organization Ontology

http://ontology.eil.utoronto.ca/icity/Organization.owl

Namespace: org

[14] http://ontology.eil.utoronto.ca/GCI/Shelters/GCI-Shelters.html

- Organization: A company or other sort of group of individuals in the urban system with some goal(s).
  An Organization may **own** Property, including different types of Buildings.
  An Organization may have an address.
  An Organization has at least 2 members.
  An Organization has some Goal(s); this represents some state or complex states, and allows for the representation of various groups' responsibilities.
  An Organization may be divided into Divisions.
- Organization Agent: Members of an organization.
  Organization Agents have goals, authority, and may be members of some team.
  An Organization Agent plays a Role within the Organization.
- Role: A Role has a single (possibly complex) Goal.
  A Role has some authority, requires some skill, and may also have some associated processes.
- Firm: A Firm is a type of organization.
  A Firm has an address and an industry type, and some Employees.
  A Firm may have a Business Establishment(s).
- Business Establishment: A Business establishment is a physical location where a Firm conducts business.
  A Business Establishment has a Location and may have an address.
- Employee: A Firm **has** some **Employees**, whom it **employs for** some Occupation.
  An Employee is a type of Organization Agent.
  An Employee may be employed at a particular Business Establishment.
  An Employee may be responsible for one or more Roles within the Organization.
  An Employee is **employed by** some Organization, unless the Person is self-employed.
  An Employee **has a Wage/Salary** and **may work at** some **Location** (this may be the location of the Firm, an alternate Location, or a Location that is subject to change).
  An employee has some employment status. An employment status may be categorized as one of: full-time regular, part-time regular, full-time-work-at-home, part-time-work-at-home
- Student: A Student is a kind of Organization Agent (and Person) who is enrolled in some EducationalInstitution
- Occupation: An occupation describes the type of work performed by some employee. Different classes of occupations may be defined, such as: General Office / Clerical, Manufacturing / Construction / Trades, Professional / Management / Technical, Retail Sales and Service

Object	Property	Value
OrganizationPD	subclassOf	change:TimeVaryingConcept
	equivalentClass	change:hasManifestation some Organization and change:hasManifestation only Organization
	change:existsAt	exactly 1 time:Interval
Organization	subclassOf	change:Manifestation
	subclassOf	tove:Organization
	equivalentClass	change:manifestationOf some OrganizationPD and change:manifestationOf only OrganizationPD
	change:existsAt	exactly 1 time:TemporalEntity
	schema:address	only schema:PostalAddress
	tove:has_goal	only tove:Goal
	tove:consists_of	only tove:Division
	spatialloc:assiociatedLocation	only geosparql:Feature
tove:Role	tove:has_goal	only tove:Goal

	tove:has_process	only (tove:Process or activity:Activity)
	tove:has_authority	only tove:Authority
	tove:requires_skill	only tove:Skill
	tove:has_resource	only resource:ResourceType
tove:Goal	subClassOf	StateType
FirmPD	subclassOf	tove:Organization
	hasFirmId	only FirmId
	equivalentClass	change:hasManifestation some Firm and change:hasManifestation only Firm
	change:existsAt	exactly 1 time:Interval
Firm	subclassOf	tove:Organization
	equivalentClass	change:manifestationOf some FirmPD and change:manifestationOf only FirmPD
	change:existsAt	exactly 1 time:TemporalEntity
	schema:address	exactly 1 schema:PostalAddress
	hasIndustryType	only IndustryType
	hasEstablishment	only BusinessEstablishment
BusinessEstablishmentPD	subclassOf	change:TimeVaryingConcept
	change:existsAt	exactly 1 time:Interval
	hasBusinessId	only BusinessId
	equivalentClass	change:hasManifestation some BusinessEstablishment and change:hasManifestation only BusinessEstablishment
BusinessEstablishment	subclassOf	change:Manifestation
	equivalentClass	change:manifestationOf some BusinessEstablishmentPD and change:manifestationOf only BusinessEstablishmentPD
	change:existsAt	exactly 1 time:TemporalEntity
	spatial_loc:hasLocation	exactly 1 spatial_loc:SpatialFeature
	schema:address	only schema:PostalAddress
tove:OrganizationAgent	tove:member_of	only tove:Division
	tove:plays	only tove:Role
	tove:has_goal	only tove:Goal
	tove:has_authority	only tove:Authority
Employee	subClassOf	tove:OrganizationAgent
	employedAs	some Occupation

	hasPay	some Wage or Salary
	worksAt	some spatial_loc:SpatialFeature
	hasEmploymentStatus	only EmploymentStatus
FullTimeEmployee	subClassOf	Employee
FullTimeHomeEmployee	subClassOf	FullTimeEmployee
FullTimeRegEmployee	(subClassOf	FullTimeEmployee) and (not FullTimeHomeEmployee)
PartTimeEmployee	subClassOf	Employee
PartTimeHomeEmployee	subClassOf	PartTimeEmployee
PartTimeTimeRegEmployee	(subClassOf	PartTimeEmployee) and (not PartTimeHomeEmployee)
Wage	hourlyPay	exactly 1 monetary:MonetaryValue
	overtimePay	only monetary:MonetaryValue
Salary	hasAnnualPay	exactly 1 monetary:MonetaryValue
tove:Activity	equivalentClass	activity:Activity
tove:Resource	equivalentClass	resource:Resource
EmploymentStatus	equivalentClass	{fulltime_regular, parttime_regular, fulltime_home, parttime_home}
GeneralOffice	subClassOf	Occupation
Trades	subClassOf	Occupation
Professional	subClassOf	Occupation
Sales	subClassOf	Occupation
EducationalInstitution	subClassOf	Organization
Student	subClassOf	OrganizationAgent
	enrolledIn	min 1 EducationalInstitution
FullTimeStudent	subClassOf	Student
PartTimeStudent	subClassOf	Student

- **hasOrgMember subPropertyOf tove:hasMember**
- **org:Organization hasOrgMember min 2 tove:OrganizationAgent**

Reused Ontologies:

- tove: The TOVE Organization ontology[15], as originally presented by (Fox, Barbuceanu, Gruninger, & Lin, 1998) with modifications to account for the difference in our representation of states, where a Goal is a subclass of StateType, and where Activities are enabled/caused by state types.
  This modification also results in the removal of the StateEmpowerment class. Note that it is possible to introduce a similar concept if required, however this would likely take the form of a property that relates an organization agent to some state-types (where the states they are empowered to take an object to, and the object itself, are described by the state type).
- icity-foundation: iCity-Foundation Ontology
- schema.org (vocabulary)

[15] http://ontology.eil.utoronto.ca/tove/organization.html

Future Work:

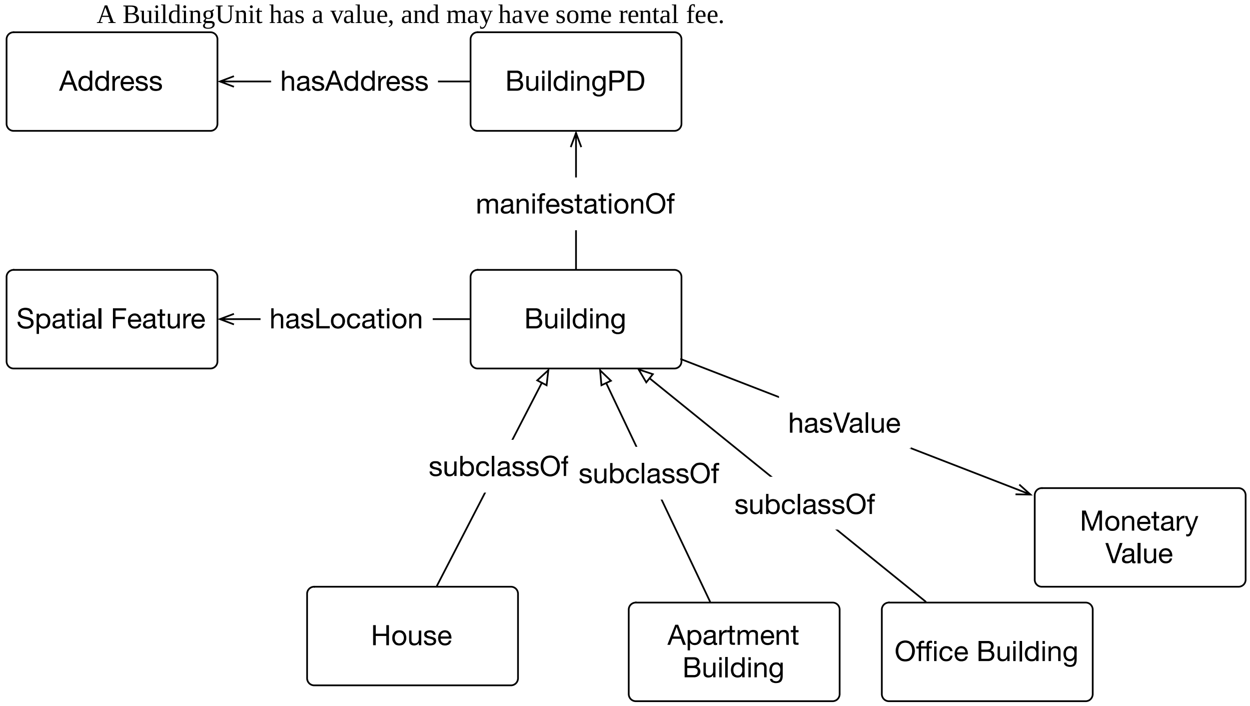
- Define part-time / full-time employees and students in more detail (e.g. with respect to their work locations).
- Define part-time /full-time students according to some enrollment criteria

6.6 Building Ontology

http://ontology.eil.utoronto.ca/icity/Building.owl

Namespace: building

- Building: A Building is a structure with some location in the urban system. The location of the Building in space may change due to construction, but the Parcel/Lot of land it is located on cannot.
  There are different types (**subclasses**) of buildings, such as House, Apartment Building, Office Building, and so on.
  A Building or BuildingUnit may contain some Facility(s), e.g. kitchen, bath, or air conditioning. Note that this is distinct from the notion of including amenities that are not a physical part of the Building (Unit), but which may be part of the Tenure.
  A Building has a market **value**.
  A Building **has** some **Location**.
  A Building contains one or many units.
- BuildingUnit: A BuildingUnit has a size (square footage, number of rooms)
  A Building or BuildingUnit may contain some Facility(s), e.g. kitchen, bath, or air conditioning. Note that this is distinct from the notion of including amenities that are not a physical part of the Building (Unit), but which may be part of the Tenure.
  A BuildingUnit has an address.
  A BuildingUnit has a value, and may have some rental fee.



Object	Property	Value
BuildingPD	subClassOf	change:TimeVaryingConcept
	equivalentClass	change:hasManifestation some Building and change:hasManifestation only Building
	change:existsAt	exactly 1 Interval

Building	equivalentClass	change:manifestationOf some BuildingPD and change:manifestationOf only BuildingPD
	subClassOf	change:Manifestation
	change:existsAt	exactly 1 TemporalEntity
	spatial_loc:hasLocation	exactly 1 spatial_loc:SpatialFeature
	monetary:hasValue	only monetary:MonetaryValue
	hasBuildingFacility	only Facility
	hasBuildingUnit	only BuildingUnit
House	subclassOf	Building
ApartmentBuilding	subclassOf	Building
OfficeBuilding	subclassOf	Building
IndustrialBuilding	subclassOf	Building
BuildingUnitPD	subclassOf	change:TimeVaryingConcept
	change:existsAt	exactly 1 Interval
	equivalentClass	change:hasManifestation some BuildingUnit and change:hasManifestation only BuildingUnit
	~~mereology:containedIn~~ unitInBuilding	exactly 1 Building
	schema:address	exactly 1 schema:PostalAddress
BuildingUnit	subclassOf	change:Manifestation
	equivalentClass	change:manifestationOf some BuildingUnitPD and change:manifestationOf only BuildingUnitPD
	change:existsAt	exactly 1 TemporalEntity
	monetary:hasValue	only monetary:MonetaryValue
	hasRent	only monetary:MonetaryValue
	hasUnitSize	only om:area
	hasRooms	only xsd:int
	~~hasFacility~~ hasBuildingFacility	only Facility

Property	Characteristic	Value (if applicable)
hasBuildingFacility	subPropertyOf	mer:hasComponent
hasBuildingUnit	inverseOf	unitInBuilding
	subPropertyOf	mer:hasComponent
	subPropertyOf	mer:contains
unitInBuilding	inverseOf	hasBuildingUnit
	subPropertyOf	mer:componentOf
	subPropertyOf	mer:containedIn

Reused Ontologies:

- Change

- Units of measure
- Mereology
- Spatial location

6.7 Vehicle Ontology

http://ontology.eil.utoronto.ca/icity/Vehicle.owl

Namespace: icity-vehicle

- Vehicle: A Vehicle provides a means of transportation within the urban system.
  A Vehicle is **associated with some Mode** of transportation.
  A Vehicle has a Vintage.
  A Vehicle has a Manufacturer (make).
  There are different types (**subclasses**) of vehicles: Motorcycle, Sedan, Truck, Bus, Commercial Cargo Vehicle, … These types may be identified and defined in different, complementary ways. The VehicleType class allows for the specifications of various types of vehicles, which may or may not also be captured as subclasses of the Vehicle class. Should a vehicle type also be a subclass, then the subclass should be defined such that it is equivalent to the class of all individuals that have the vehicle type as a property *hasVehicleType value <vehicle type>*.
  A Vehicle **has a capacity** of passengers
  A Vehicle **has a capacity** of cargo
  A Vehicle **has a Speed** at some point in time
  A Vehicle **has a location** at some point in time.

Object	Property	Value
VehiclePD	equivalentClass	change:hasManifestation some Vehicle and change:hasManifestation only Vehicle
	subclassOf	change:TimeVaryingConcept
	change:existsAt	exactly 1 time:Interval
	hasVehicleType	only VehicleType
	schema:productionDate	only time:DateTimeDescription
	schema:brand	only schema:Brand
	schema:vehicleSeatingCapacity	exactly 1 xsd:int
	schema:cargoVolume	only om:volume
	hasCargoCapacityLoad	only om:Quantity
	schema:driveWheelConfiguration	schema:DriveWheelConfigurationValue
	schema:fuelConsumption	schema:QuantitativeValue
	schema:fuelEfficiency	schema:QuantitativeValue
	schema:fuelType	schema:QualitativeValue
	schema:mileageFromOdometer	schema:QuantitativeValue
	schema:numberOfDoors	only xsd:int
	schema:numberOfAxels	only xsd:int
Vehicle	equivalentClass	change:manifestationOf some VehiclePD and

		change:manifestationOf only VehiclePD
	subclassOf	change:Manifestation
	change:existsAt	exactly 1 time:TemporalEntity
	schema:purchaseDate	only time:DateTimeDescription
	hasSpeed	only om:speed
	spatial_loc:hasLocation	only spatial_loc:SpatialFeature
	accommodatesWheelchair	max 1 xsd:Boolean
	accommodatesBicycle	max 1 xsd:Boolean
schema:QualitativeValue	subClassOf	om:quantity

Ontologies Reused:

- Schema.org (vocabulary)
- iCity-Foundation

6.8 Transportation System Ontology

http://ontology.eil.utoronto.ca/icity/TransportationSystem.owl

Namespace:transport

While most existing work attempts to describe the network based on its physical constructs, we model the network flow and the physical infrastructure separately. The motivation for this is that the constraints on transportation flow are something that is *applied to* the physical infrastructure. These constraints are distinct from the physical characteristics and so should be defined separately. Although some constraints may be related, such as flow constraints imposed by the size of the lane that an arc accesses, this is a specific relationship that should be captured rather than conflating the concepts. For example, there is nothing to stop a vehicle from going the wrong way on a road, except for the flow of traffic that is imposed on the system (and these constraints may change with time). This results in the identification of two key concepts: the Transportation Network (a directed graph), and the Transportation Complex (a physical feature where transportation occurs).

We relate the Network and the Infrastructure by relating an Arc to a Transportation Complex (or other Road Segment) with the "accesses" property. In this way, we may define an Arc accessing various Transportation Complexes at different Levels of Detail (LOD).
Both Nodes and Arcs may have implicit locations based on the infrastructure they access, however unlike the infrastructure classes, Nodes and Arcs are *not* Spatial Things. A Node may have a control (e.g. a signal) with a physical presence somewhere else (traffic lights apply to one side of the intersection, but are actually located on the other side of the intersection); by separating the physical infrastructure and the network flow we are able to accurately represent this.

The OTN (Ontology of Transportation Networks[16]) ontology, as presented by (Lorenz, Ohlbach, & Yang, 2005), also defines terms such as nodes, arcs, and road/rail elements. The lack of maintenance and activity on the OTN poses a potential issue, and the lack of modularity in its

[16] http://www.pms.ifi.lmu.de/rewerse-wga1/otn/OTN.owl

structure makes it difficult to use. Therefore, although its scope is similar, we have elected not to reuse it in the design of this ontology.

- Network: A collection of Nodes and Arcs that enables transportation. A Network may have some cost associated to its access.
- Link: A directed connection in the Network that enables transportation via some Mode(s) from one Node to another.
  A link contains one or more Arcs that represent individual flows of traffic (e.g. traffic lanes, bicycle lanes).
  A link begins and ends at a source and sink Node.
  A link has some (straight-line) length description, in km.
  A link is associated with, or considered to be *in*, a municipality and a planning district.
  A link supports one or more Mode(s) of access.
- Arc: A directed connection in the Network that enables transportation via a particular Mode(s) from one Node to another.
  An Arc begins and ends at the source and sink of the Link it is contained in.
  An Arc has access to some Spatial Thing (such as a road), which may change over time.
  An Arc may impose access restrictions (for example, based on the size of vehicle), which are subject to change.
  An Arc may have some cost associated to its travel.
  An Arc supports one or more Modes of access.
  An Arc may have some posted and/or free flow speed. It may also be described with a volume delay function (VDF).
- Node**:** A point in the Network at which Arcs are connected. A node as a unique identifier; for example, as defined in the EMME NCS11.
  A Node may contain different types of controls: Network Transfer, Signal Control, and Flow Control.
  A Node may be associated with specific location information (e.g. coordinates). Note that this may be subject to change. The physical location of a node (generally larger than a single point) may be inferred based on the locations of the transportation complexes which it connects.
  A Node accesses some TransportationComplex, such as an Intersection. In the future, it may be useful to define other specific types of TransportationComplexes that are accessed by nodes, (e.g. bus stops).
- Network Transfer: Enables transfer between networks at a given Node.
- Signal Control: Controls the flow of transportation between some of the incoming and outgoing arcs that the Node connects. Signal Controls have specialized attributes such as the number of phases, phase length, signal timing, type of signal. Note that the phases and/or the phase length may vary as a function of time of day or other triggers (e.g. ground sensors, traffic sensors).
- Flow Control: Controls the flow of traffic at a given Node.
  A Flow Control may be operative/inoperative at different times. For example, "no left turns from 4-6pm".
  A Flow Control may be a generalization of Signal Control.

- Mode: A mode of transportation is a **means of** performing travel within the urban system. There are various types (instances) of Mode: Foot, Bike, PersonalVehicle, PublicTransit, Cab, CommercialVehicle, Plane, Boat, Train.
- LoopDetector: A Loop Detector is a kind of Sensor that detects vehicle presence at some point on a road segment. A Loop Detector is owned by some Organization; it has some location, and is associated with (has a feature of interest) the particular part of the transportation network (i.e. a transport:Arc) that it is located on.
  A Loop Detector makes observations about the vehicle presence on the road segment that is its feature of interest.
  The vehicle presence is a proxy for the occupancy of the road segment and the average vehicle speed on the road segment.
- TTI
- MeanTTI

The physical Infrastructure of the transportation system is defined, as required, at different levels of detail (LOD). Specific types of Transportation Complex (a term we adopt from the CityGML schema) may be defined according to the Arcs that access them. We define the following types of Transportation Complex.

- Road
- Rail
- Waterway
- Airway
- Bike Trail
- Footpath
- Parking

Each Transportation Complex may be further defined as follows:

- **Road**: An aggregation of Road Segments with the same name.
- **RoadSegmentPD**: accessed only by Links that are not accessible by water or air modes. Different RoadSegments Perdurants will be accessed by Arcs that are accessible by various other Modes, not necessarily *everything* else. A Road Segment Perdurant is comprised of Road Segments that exist over time.
- **RoadSegment**: A RoadSegment has variant attributes.
  A RoadSegment has an owner, access restrictions, and is accessed by some Arc(s) -- all of which may change over time.
  A RoadSegment has some location, which is co-located with (contains the locations of) the Arcs and Nodes it contains.
- **Rail**: An aggregation of Rail Segments with the same name.
- **RailSegmentPD**: Accessed only by Arcs that are accessible by rail modes.
  A RailSegment Perdurant has an invariant location, which is co-located with (contains the locations of) the Arcs and Nodes it contains. A Rail Segment Perdurant is comprised of Rail Segments that exist over time.
- **RailSegment**: A RailSegment has an owner, access restrictions, and is accessed by some Link(s).

- Note that the location of a RoadSegment is variable (e.g. road widening or other activities do not change the identity of the road element), whereas a RailSegment's is not.
- **IntersectionPD:** Accessed only by NodePDs. An Intersection Perdurant captures the physical entity of an intersection, which is co-located with various other transportation complexes (e.g. roads, paths) that pass through it. An Intersection Perdurant is comprised of Intersections that exist over time.
- **Intersection:** An Intersection exists at some time. It has some location. It may have some owner and is accessed by some Node. In the future, it may be useful to extend this class and relate it to certain aspects of the physical infrastructure such as signs, signals, etc.

Classes may be defined for footpaths, bicycle lanes/trails, and so on. Should it be useful, this representation could be extended to define individual traffic lanes, (e.g. the transportation complex that is accessed by a single arc).

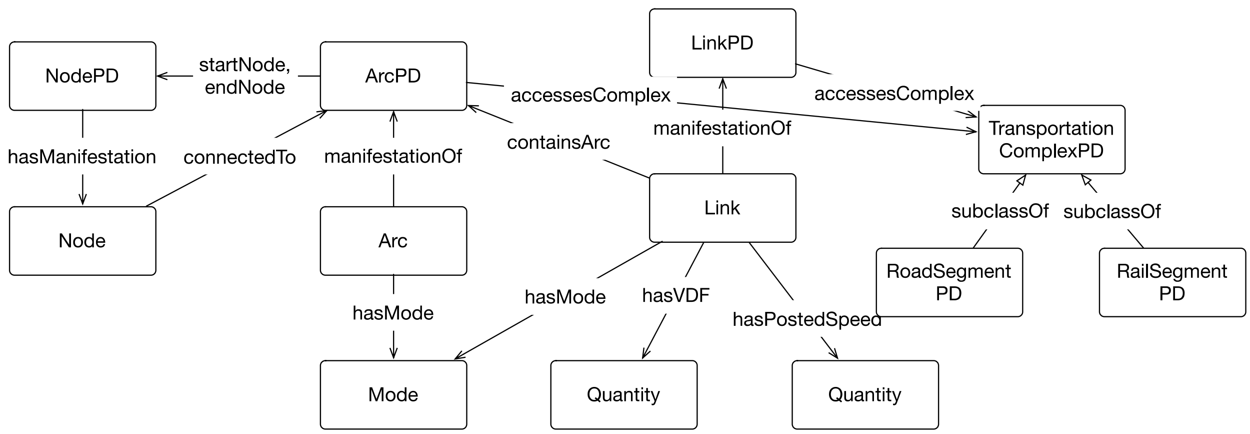


Figure 8: Structure of the Transportation Network (some omissions).

Object	Property	Value
NetworkPD	subclassOf	change:TimeVaryingConcept
	equivalentClass	change:hasManifestation some Network and change:hasManifestation only Network
	change:existsAt	exactly 1 time:Interval
Network	subclassOf	change:Manifestation
	equivalentClass	change:manifestationOf some NetworkPD and change:manifestationOf only NetworkPD
	change:existsAt	exactly 1 time:TemporalEntity
	mereology:hasComponent	only Arc or Node
NodePD	subclassOf	change:TimeVaryingConcept
	equivalentClass	change:hasManifestation some Node and change:hasManifestation only Node
	change:existsAt	exactly 1 time:Interval
	hasNodeID	max 1 NodeId

Node	subclassOf	change:Manifestation
	equivalentClass	change:manifestationOf some NodePD and change:manifestationOf only NodePD
	change:existsAt	exactly 1 TemporalEntity
	mereology:componentOf	only Network
	connectedTo	min 1 Arc
	hasControl	only (NetworkTransfer or SignalControl or FlowControl)
	associatedLocation	only spatial_loc:Feature
LinkPD	subclassOf	change:TimeVaryingConcept
	equivalentClass	change:hasManifestation some Link and change:hasManifestation only Link
	change:existsAt	exactly 1 time:Interval
	mereology:componentOf	only Network (variant or invariant?)
	startNode	exactly 1 NodePD
	endNode	exactly 1 NodePD
	accessesComplex	only TransportationComplexPD
Link	subclassOf	change:Manifestation
	equivalentClass	change:manifestationOf some LinkPD and change:manifestationOf only LinkPD
	change:existsAt	exactly 1 time:TemporalEntity
	containsArc	min 1 ArcPD
	associatedLinkLength	exactly 1 om:length
	supportsMode	min 1 Mode
	hasNumLanes	exactly 1 xsd:integer
	hasVDF	max 1 om: Quantity
	hasLinkCapacity	max 1 (om:Quantity and om:'has value' only (om:'has unit' only (om:'has numerator' only om:CardinalityUnitPerTime) and (om:'has denominator' only (om:'Cardinality Unit' and inverse(om:'has unit') only (inverse(om:'has value') only (gci:cardinality_of only (gci:defined_by only Arc)))))))
	hasFreeFlowSpeed	max 1 om:speed
	hasPostedSpeed	max 1 om:speed
	hasToll	only MonetaryValue
	inMunicipality	exactly 1 Municipality
	inPlanningDistrict	exactly 1 PlanningDistrict

ArcPD	subclassOf	change:TimeVaryingConcept
	equivalentClass	change:hasManifestation some Arc and change:hasManifestation only Arc
	startNode	exactly 1 NodePD
	endNode	exactly 1 NodePD
	change:existsAt	exactly 1 time:Interval
	accessesComplex	only TransportationComplexPD
	containedInLink	exactly 1 LinkPD
Arc	subclassOf	change:Manifestation
	equivalentClass	change:manifestationOf some ArcPD and change:manifestationOf only ArcPD
	change:existsAt	exactly 1 time:TemporalEntity
	accessesComplex	only TransportationComplex
	mereology:componentOf	only Network
	hasControl	only AccessRestriction
	supportsMode	min 1 Mode
	hasLaneCapacity	exactly 1 om:CapacityRate
	hasVDF	max 1 om:quantity
	hasFreeFlowSpeed	max 1 om:speed
	hasPostedSpeed	max 1 om:speed
	hasToll	only MonetaryValue
	inMunicipality	exactly 1 Municipality (?) tbd – where should municipalities be defined
	inPlanningDistrict	exactly 1 PlanningDistrict
NetworkTransfer	controlFor	only Node
	connectsNetworks	min 2 Network
FlowControl	controlFor	only Node
	hasInflow	min 1 Arc
	hasOutflow	min 1 Arc
SignalControlPD	subClassOf	change:TimeVaryingConcept
	equivalentClass	change:hasManifestation some SignalControl and change:hasManifestation only SignalControl
	change:existsAt	exactly 1 time:Interval
	controlFor	only Node
	hasInflow	min 1 Arc
	hasOutflow	min 1 Arc
SignalControl	subClassOf	change:Manifestation
	equivalentClass	change:manifestationOf some SignalControlPD and change:manifestationOf only SignalControlPD
	change:existsAt	exactly 1 time:TemporalEntity

	hasPhase	only SignalPhase
SignalPhase	signalLength	only time:DurationDescription
TransportationComplexPD	subClassOf	change:TimeVaryingConcept
	equivalentClass	change:hasManifestation some TransportationComplex and change:hasManifestation only TransportationComplex
TransportationComplex	subclassOf	change:Manifestation
	equivalentClass	change:manifestationOf some TransportationComplexPD and change:manifestationOf only TransportationComplexPD
	spatial_loc:hasLocation	only spatial_loc:Feature
otn:Road	hasRoadId	only RoadId
	aggregationOf	only RoadSegment
RoadSegmentPD	subclassOf	TransportationComplexPD
	equivalentClass	change:hasManifestation some RoadSegment and change:hasManifestation only RoadSegment
	hasRoadSegmentId	only RoadSegmentId
	change:existsAt	exactly 1 time:Interval
RoadSegment	equivalentClass	otn:RoadElement
	subClassOf	TransportationComplex
	equivalentClass	change:manifestationOf some RoadSegmentPD and change:manifestationOf only RoadSegmentPD
	change:existsAt	exactly 1 time:TemporalEntity
	spatial_loc:hasLocation	only spatial_loc:SpatialFeature
Mode	equivalentClass[17]	{C,E,F,H,I,J,B,G,L,M,P,Q,R,S,A,K,T,U, V,W,Y}
Municipality		
PlanningDistrict		
LoopDetector	sosa:detects	{vehicle_presence}
	sosa:observes	{road_occupancy}
	sosa:observes	{vehicle_volume}
	sosa:observes	{mean_travel_speed}
	sosa:madeObservation	only (sosa:Observation and sosa:hasFeatureOfInterest only transport:Arc and sosa:wasOriginatedBy

[17] More options may be added as required. This list comes from the options specified in the EMME NCS11.

		{vehicle_presence} and sosa:hasResult RoadOccupancy or VehicleVolume or MeanTravelSpeed)
{vehicle_presence}	a	ssn:Stimulus
{road_occupancy}	a	ssn:ObservableProperty
{vehicle_volume}	a	ssn:ObservableProperty
{mean_travel_speed}	a	ssn:ObservableProperty
VehicleVolume	subClassOf	om-2:Quantity
	om-2:hasValue	only (om-2:hasUnit only CardinalityUnitPerTime)
	gci:cardinalityOf	only LocVehiclePopulation
LocVehiclePopulation* *precise definition only possible for a particular location	gci:definedBy	only (Vehicle and hasLocation some Feature)
RoadOccupancy	subClassOf	om-2:Quantity
	om-2:hasValue	only (om-2:hasUnit only RoadOccupancyUnit)
RoadOccupancyUnit	subClassOf	om-2:UnitDivision
	om-2:hasNumerator	only om-2:TimeUnit
	om-2:hasDenominator	only om-2:TimeUnit
MeanTravelSpeed	subClassOf	om-2:Speed
	om-2:hasAggregateFunction	value {om-2:average}
LaneCapacity_unit	subClassOf	om-2:Unit
LinkCapacity_unit	subClassOf	om-2:Unit
{vehicles_per_hour}	a	LaneCapacity_unit
{vehicles_per_hour_per_lane}	a	LinkCapacity_unit

1. Note that the classes of observable properties are primarily introduced for consistency with the SSN representation as a means of capturing the semantics of a class of Sensors (in this case, Loop Detectors). Any instance of, e.g. RoadOccupancy simply corresponds to a RoadSegment occupied by some thing, or occupied by nothing:
   RoadOccupancy(x) ⇔ isPropertyOf(x,y) & RoadSegment(y) & [exists (t) occupiedBy(y,t) | -exists(t) occupiedBy(y,t)]
   As a consequence of the 4D representation, an instance of the observable property RoadOccupancy refers to a property of a road segment at some time, t.
2. Additional semantics of sensors and observations: temporal restrictions, connection to object properties
3. A RoadSegment's vehicle count can be calculated based on the KB, but can we formalize the relationship between the count and the KB? i.e. number of observations in a given interval?

IntersectionPD	subclassOf	change:TimeVaryingConcept
	equivalentClass	change:hasManifestation some Intersection and change:hasManifestation only Intersection

	inverse(accessesComplex)	only NodePD
	change:existsAt	exactly 1 time:Interval
Intersection	equivalentClass	otn:RoadElement
	subclassOf	change:Manifestation
	subClassOf	TransportationComplex
	equivalentClass	change:manifestationOf some RoadSegmentPD and change:manifestationOf only RoadSegmentPD
	change:existsAt	exactly 1 time:TemporalEntity
	spatial_loc:hasLocation	only geosparql:Feature
	inverse(accessesComplex)	only Node

Ontologies Reused:
- Change
- SpatialLoc
- SSN: Semantic Sensor Network ontology to capture sensors and their observations. These observations are processed or used directly as attributes of the network.

Notes:
- We observe that the properties *inMunicipality* and *inPlanningDistrict* may apply to other areas of the domain (e.g. land use, building ontologies), in which case they will be better defined at a lower (more foundational) level within the ontology. However, as they are currently only required for the Transportation System sub-ontology, it is currently not clear where and how this should be done. For now, we define these properties within the Transportation Network System ontology and leave the final organization for a future iteration if and when requirements for their widespread use are identified.

Future Work:
- Define lane and link capacity units in greater detail (e.g. with numerators and denominators).
- There is a relationship between the modes of access of a link and those of the arcs it contains that should be captured in a more detailed representation.

6.8.1 Travel Costs

http://ontology.eil.utoronto.ca/icity/TravelCost.owl

Namespace: icity-travelcost

An extension of the transportation network (and other generic ontologies) is required in order to represent the different costs associated with accessing and travelling on the networks. These may take the form of direct costs such as tolls and fares, or possible indirect costs such as vehicle wear and tear, gas, etc. In addition, there may be non-monetary costs associated with travel such as pollution and travel time. Costs are associated with Network access, but also with individual Arcs. They may also be dependent on situational factors such as time of day, or age of traveler. Travel Costs define the costs associated with accessing the transportation system; a travel cost is a property of an arc or its network. We define a separate extension of Trip Costs to capture other, indirect costs that may vary between individual trips; a trip cost is a property of some instance of travelling.

- Travel Cost: There are different types of Travel Costs which are derived from different factors, and may be defined in different ways. Travel Costs apply to Arcs and / or Networks.
- Distance Fee is a type of Travel Cost
  Distance Fee has an associated Cost
  It applies for a certain distance (between nodes, or per km)
  It applies to some Arc

It may have an associated time-of-day applicability
It may be associated to specific modes of transport

- Access Fee is a type of Travel Cost
  Access Fee has an associated Cost
  It may have an associated time-of-day applicability
  It may be associated to specific modes
  It applies to some Network

Object	Property	Value
TravelCost	travelCostOf	only (transportation:Arc or transportation:Network)
	applicableFor	only time:TimePeriod or time:CalendarPeriod
	applicableTo	only transportation:Mode
	hasMonetaryCost	only monetary:MonetaryValue
transportation:Arc	hasTravelCost	only TravelCost
transportation:Network	hasTravelCost	only TravelCost
DistanceFee	subclassOf	TravelCost
	forDistance	only om:length
	travelCostOf	only transportation:Arc
AccessFee	subClassOf	TravelCost
	travelCostOf	only transportation:Network

Property	Characteristic	Value (if applicable)
travelCostOf	inverseOf	hasTravelCost

Ontologies Reused:

- iCity-Transportation Network

6.9 Parking Ontology

http://ontology.eil.utoronto.ca/icity/Parking.owl

Namespace: parking

- Parking Area: Parking Area refers to some area that enables parking of Vehicles.
  A Parking Area may contain **sub-Parking Areas**, the area of which may change.
  A Parking Area has some Parking **Policy**
  A Parking Area may provide **car changing** stations.
  A Parking Area has some **Location**.
  A Parking Area has some vacancy (or occupancy) at some point in time.
  A parking area may be owned by some Person or Organization, and it may be allocated for some Building, Location, Person, or Organization. Note that ownership and allocation of a Parking Area are distinct: an organization may own a parking area, but it may be allocated (e.g. rented) to some other organization or individual(s).
  A Parking Area may have some hours of operation.
  A Parking Area may have some limit on the dimensions of allowed vehicles (height/width/length)
  associated location information (e.g. nearby crossroads, landmark, etc)
  Different types (subclasses) of Parking Area may be defined as required, such as Street Parking Area, Lot Parking Area, Garage Parking Area, Illegal Parking Area, Loading/Unloading Zone Parking Area,…
- Parking Space: A Parking Space is a Parking Area with the capacity for a single vehicle. (hasCapacity 1, hasVacancy 0 or 1). Specializations of parking space may be defined based on accommodated vehicle type (e.g. small vehicles, commercial vehicles, electric vehicles,…).
  has Reservations?
  has Schedules?

A Parking Space may or may not be occupied by some vehicle at a particular point in time. If a space is occupied, its availability may be determined (or approximated) based on the scheduled/purchased time by its current occupant.

- A Parking Facility is a parking area that is not contained by any other parking area. A Parking Facility may be owned by some organization and have some hours of operation; it may have a name, contact phone number, address, and possibly an associated website.
- Accessible Parking Space: A type of parking space reserved for users with disabilities
- EV Space: A type of parking space that provides access to some EV Charger(s)
- Parking Policy: A Parking Policy dictates under what terms some Parking Area is accessible for parking.
  A Parking Policy may have a **Rate**.
  A Parking Policy may have a **max duration**.
  A Parking Policy may have **allowable periods** *(these periods must be during the hours of operation of the parking area)*.
  A Parking Policy may apply only to a particular class of users.
  Different sorts of parking policies (subclasses) may be defined: e.g. free parking, policies for EVs, persons with accessibility needs.
- Rate: A Rate has a **monetary value** and an **associated duration**.
  A Rate has a **ParkingPaymentMethod** (e.g. mobile, license plate entry, cashier, meter).
  A Rate may have some minimum charge, specified as either a monetary value or duration (e.g. regardless of the time parked, the customer will be charged at least $5, or the rate will be applied for at least 30 min).
  A maximum cost may also be specified; for example, the rate may be $5 per hour, with a maximum of $20 to park for the remainder of the policy's hours of operation. It is not always the case that the maximum cost coincides with the maximum time-based rate of the hours parked.
- EV charger: A charger for electric vehicles is an amenity which may be provided by some parking spaces.
  An EV charger has some model and is capable of charging certain classes of vehicles.
  An EV charger may be available or unavailable at a given time. This availability may be predetermined based on the scheduled duration of a vehicle's occupancy, and the time left to charge the vehicle.

Object	Property	Value
ParkingAreaPD	subclassOf	change:TimeVaryingConcept
	equivalentClass	change:hasManifestation some ParkingArea and change:hasManifestation only ParkingArea
	change:existsAt	exactly 1 time:Interval
	spatial_loc:hasLocation	exactly 1 spatial_loc:SpatialFeature
	spatial_loc:hasAssociatedLocation	only spatial_loc:SpatialFeature
	maxAdmittableHeight	exactly 1 om:length
	maxAdmittableWidth	exactly 1 om:length
	maxAdmittableLength	exactly 1 om:length
	has Address	only icontact:Address
ParkingArea	subclassOf	change:Manifestation
	equivalentClass	change:manifestationOf some ParkingAreaPD and change:manifestationOf only ParkingAreaPD
	change:existsAt	exactly 1 time:TemporalEntity
	hasSubParkingArea	only ParkingArea

	hasVehicleCapacity	only (CapacitySize and gci:cardinality_of only (gci:defined_by only Vehicle))
	hasParkingPolicy	only ParkingPolicy
	hasChargingStations	exactly 1 xsd:integer
	resource:ownedBy	some Person or Organization
	occupiedBy	only Vehicle
	isOpen	exactly 1 xsd:boolean
	hasParkingService	only ParkingService
	parkingAllocatedTo	only (Person or Building or Organization or Feature)
ParkingFacilityPD	subclassOf	park:ParkingAreaPD
	equivalentClass	change:hasManifestation some ParkingLot and change:hasManifestation only ParkingLot
ParkingFacility	subClassOf	ParkingArea
	subParkingAreaOf	exactly 0 ParkingArea
	foaf:name	only xsd:string
	icontact:hasWebsite	only xsd:string
	icontact:hasAddress	only contact:Address
	icontact:hasOperatingHours	only rec:HoursOfOperation
	icontact:hasTelephone	only icontact:PhoneNumber
ParkingSpace	subclassOf	ParkingArea
	hasVehicleCapacity	some (om:hasValue some (om:has_numerical_value value 1))
AccessibleSpace	subclassOf	ParkingSpace
	hasParkingPolicy	only AccessibilityParkingPolicy (to define)
EVSpace	subclassOf	ParkingSpace
	hasParkingPolicy	only EVParkingPolicty (to define)
ParkingService	*may be defined in greater detail in the future	
Valet	subclassOf	ParkingService
Carwash	subclassOf	ParkingService
ParkingPolicy	hasParkingRate	only ParkingRate
	maxDuration	only time:DurationDescription
	appliesDuring	only contact:HoursOfOperation
	appliesTo	only person:Person
	appliesFor	only vehicle:VehicleType
	hasGracePeriod	max 1 time:DurationDescription
	excludesPublicHoliday	exactly 1 xsd:boolean
ParkingRate	hasMonetaryCost	only om:MonetaryValue
	forDuration	only time:DurationDescription
	hasPayment	only ParkingPaymentMethod

	appliesTo	only person:Person
	minParkingCharge	only (om:MonetaryValue or time:DurationDescription)
	maxParkingCost	only om:MonetaryValue
FreeParkingPolicy	hasParkingRate	only (ParkingRate and hasMonetaryCost only (om:MonetaryValue and om:numerical_value {0}))

Property	Characteristic	Value (if applicable)
hasSubParkingArea	subPropertyOf	mer:hasProperPart
	domain	ParkingArea
	range	ParkingArea
	inverse	subParkingAreaOf
subParkingAreaOf	subPropertyOf	mer:properPartOf
	domain	ParkingArea
	range	ParkingArea
	inverse of	hasSubParkingArea

Ontologies Reused:

- Mereology
- Change
- Time
- OM
- Person
- Vehicle
- Contact

Future work:

Constraints may be defined to relate the hours of operation with the parking lot's associated parking policies and their hours of operation: a parking lot should have policies defined during all of its hours of operation.

If required, parking services may be defined in greater detail.

6.10 Public Transit Ontology

http://ontology.eil.utoronto.ca/icity/PublicTransit.owl

Namespace: transit

- TransitSystem: A TransitSystem is a **collection of Routes**.
  A TransitSystem may be **accessed by** some Fare or Transit Pass.
- Route: A Route consists of a series of Route Links and may be divided into Route Sections.
  A Route has some directionality (captured by the route links).
- Route Section: A Route Section is part of some Route and consists of Route Links.
  A Route Section begins and ends at a Stop Point.
- Route Link: A Route Link is part of some Route. It is a primitive element of a route, operating on single Arc or Link within the transportation system.
- Stop Point: A Stop Point marks the **start or end of** a Route Link (e.g. a subway stop or bus stop).
  A Stop Point is a subclass of a Node, as defined in the Transportation System ontology.

Like a Node, a Stop Point **has an associated Location**.
A Person may enter or exit the transit vehicle at a Stop Point.
(to do: Station subclass of StopPoint)

- Station: A Station is a specialized type of Stop Point that contains multiple Stop Points.
  A Station may have an actual and associated location; it may provide various amenities (e.g. businesses or restrooms)
- Transit Incidents, broadly, are events of interest that occur on a particular transit trip. Typically, they are problematic, unplanned issues resulting in some delay.
  A TransitIncident is a subclass of Activity.
  It is associated with some station or stop point.
  An incident may be described (and so classified) by a predefined code: hasCode only xsd:String.
  An incident will have some resulting caused gap (i.e. the time from the incident until the next train arrives at the station).
- TransitTrip is a subclass of Trip.
  Transit Trips have specific restrictions and specialized properties. A Transit Trip occurs on some predefined route. A Transit Trip may also describe a trip on some smaller part of a Route, i.e. a Route Link. In exceptional cases, is possible that a TransitTrip may occur off-route (e.g. detours). The start and destination of a Transit Trip must be a Stop Point, and all Transit Trips must be performed with a Transit Vehicle.
- ScheduledTransitTrip is a type of RecurringEvent that only has TransitTrips as occurrences. A ScheduledTransitTrip is scheduled on some Route, RouteLink, or RouteSection, however it is not necessarily the case that the trip is accessible to travelers at the beginning stop point. It is possible that the scheduled trip will not pick up any passengers, or that passengers must pre-arrange in order to be picked up by the scheduled trip. A Scheduled Transit Trip may have a pick-up type and/or drop-off type as defined by some Trip Access Arrangement Type: as scheduled, not available, arranged with agency, or arranged with driver.
  ScheduledTransitTrips may be used to specify route and stop timetables. Like a TransitTrip, a ScheduledTransitTrip may be described as inbound or outbound with the isOutbound data property.
  Scheduled trips may be defined to require only the assignment of vehicles that accommodate a wheelchair rider(s); this property may be captured with the isWheelchairAccessible data property.
  The start and end times of scheduled (recurring) transit trips may be used to specify route and stop timetables.
- TransitVehicle is a subclass of Vehicle.
  A TransitVehicle has a transit vehicle id. This refers to the identifier assigned by the transit authority, as opposed to a serial number.
  Transit Vehicles are owned and operated by some transit authority. There are specialized types of transit vehicles (e.g. different types of streetcars), and a restricted set of modes. Transit Vehicles typically only operate on pre-defined routes, however there are exceptions (e.g. detours, travel for maintenance, etc).
- AccessMethod: An Access Method is the means of **access to** a Line
  An AccessMethod **has a Monetary Value**.
  An AccessMethod may be **valid for** a specific distance or time.
- RouteTimetable: A Timetable represents schedule information for a particular Route, or Route Link.
  A RouteTimetable has an **expected travel time (Duration)** for the Route, or Route Link.
- A StopTimetable has an **expected arrival time (Time Instant)** for some Stop Point.
- VehicleBlock: A Vehicle Block represents a grouping of transit trips to be allocated to a particular vehicle. A transit trip is part of a single block and each block may contain multiple transit trips, therefore the allocatedFor property relating vehicle blocks and transit trips is inverse functional. Each block may be allocated multiple vehicles, but only one vehicle at a given point in time therefore the allocatedTo property which relates vehicle blocks to vehicles is functional.
- Two complementary properties (one object and one data property) have been added to capture information regarding transit passes. The data property provides a simply Boolean value to capture whether a person (at some time) has a transit pass; whereas the object property provides the ability to associate a particular transit pass (with some properties regarding, for example, its access, cost, and balance).

Object	Property	Value

TransitSystemPD	subclassOf	change:TimeVaryingConcept
	equivalentClass	change:hasManifestation some TransitSystem and change:hasManifestation only TransitSystem
	change:existsAt	exactly 1 time:Interval
	operatedBy	org:OrganizationPD
TransitSystem	subclassOf	change:Manifestation
	equivalentClass	change:manifestationOf some TransitSystemPD and change:manifestationOf only TransitSystemPD
	change:existsAt	exactly 1 time:TemporalEntity
	hasRoutes	only Route
	accessBy	only AccessMethod
AccessMethod	hasMonetaryCost	only monetary:MonetaryValue
	validFor	only (time:DurationDescription or om:length)
Fare	subclassOf	AccessMethod
TransitPass	subclassOf	AccessMethod
RoutePD	~~hasRouteId~~	~~exactly 1 RouteId~~
	subClassOf	change:TimeVaryingConcept
	equivalentClass	change:hasManifestation some Route and change:hasManifestation only Route
	change:existsAt	only time:Interval
	hasGTFSRouteType	exactly 1 {0,1,2,3,4,5,6,7}
Route	subclassOf	change:Manifestation
	equivalentClass	change:manifestationOf some RoutePD and change:manifestationOf only RoutePD
	change:existsAt	only time:TemporalEntity
	routeShortName	max 1 xsd:string
	foaf:name	max 1 xsd:string
	hasSection	only RouteSection
	operatesOn	only ArcPD
	hasDisplayColor	max 1 xsd:string
	hasRouteTextColor	max 1 xsd:string
	icontact:hasOperatingHours	some rec:HoursOfOperation
RouteSection	mereology:contains	only RouteLink
	beginsAtStop	exactly 1 StopPoint
	endsAtStop	exactly 1 StopPoint
	operatesOn	only ArcPD

RouteLink	operatesOn	exactly 1 ArcPD
StopPoint	subclassOf	transport:Node
	spatial_loc:hasLocation	exactly 1 spatial_loc:Feature
	transit:hasStopCode	exactly 1 xsd:string
	foaf:name	min 1 xsd: string
	transit:wheelchairBoarding	exactly 1 xsd:boolean
AccessibleStopPoint	equivalentClass	StopPoint and transit:wheelchairAccessible value true
Station	subclassOf	StopPoint
	mereology:contains	min 1 StopPoint
	spatial_loc:associatedLocation	some spatial_loc:Feature
TransitIncident	subclassOf	activity:Activity
	associatedWithStop	only StopPoint
	hasIncidentCode	min 1 xsd:string
	causedGap	only time:Interval
	associatedWithTrip	only TransitTrip
TransitTrip	subclassOf	trip:Trip
	transit:occursOn	only transit:Route or transit:RouteSection or transit:RouteLink or transport:TransportationComplex
	transit:viaVehicle	exactly 1 transit:TransitVehicle
	transit:isOutbound	only xsd:boolean
ScheduledTransitTrip	subclassOf	rec:RecurringEvent
	rec:hasOccurrence	only transit:TransitTrip
	transit:scheduledOn	only transit:Route or transit:RouteSection or transit:RouteLink
	transit:isOutbound	only xsd:boolean
	transit:isWheelchairAccessible	only xsd:boolean
	hasPickupType	max 1 TripAccessArrangement
	hasDropoffType	max 1 TripAccessArrangement
TripAccessArrangement	equivalentClass	{AccessAsScheduled, AccessNotAvailable, AccessArrangedViaAgency, AccessArrangedViaDriver}
TransitVehicle	subclassOf	vehicle:Vehicle
	hasTransitVehicleId	exactly 1 xsd:string
VehicleBlock	assignedTo	only transit:TransitVehicle
	assignedFor	min 1 ScheduledTransitTrip
person:Person	transitPass	only TransitPass
	hasTransitPass	only xsd:boolean

Ontologies Reused:

- Activity

- Change
- Spatial Location
- TransportationSystem
- Trip
- Organization

Future work:

Though not applicable for the TTC, future work should consider a representation of zone or similar information that may be used in some systems to calculate fare cost.
There is some potential to incorporate detailed constraints on the types of routes (bus, rail, etc) and the arcs in the network that the routes access, according to the mode supported by the arcs. Constraints may also be enforced on the times of trips as compared to the hours of operation for a particular route (i.e. a trip should occur within the defined hours of operation).
With additional information, the stops associated with a particular trip may be validated against its direction id to confirm that the sequence is capturing either an inbound or outbound path.
Constraints may be added to enforce the types of vehicles that perform a particular transit trip, based upon the specifications of the scheduled trip of which the transit trip is an occurrence. For example, if the scheduled trip is wheelchair accessible, then any vehicle that performs the transit trips (or is assigned a block containing the scheduled trip) should accommodate a wheelchair. On the other hand, it may be the case that vehicle assignments sometimes conflict with the scheduled trip type and so such constraints may not be accurate/desirable.
We may also be able to infer whether a stop offers wheelchair boarding based on the associated routes and trips.
Rules may be added to express the relationship between the Arc that a Route Link operates on and the set of Arcs that a Route Section or Route operate on (i.e. the sum of all Arcs operated on by all Route Links contained in the Route Section/Route).
In extensions beyond OWL, it may be useful to formalize the relationship between transitPass and hasTransitPass properties.

6.11 Land Use Ontology

http://ontology.eil.utoronto.ca/icity/LandUse

Namespace: landuse

- Parcel: A Parcel is a way of defining some area in an urban system.
  A Parcel **has a Location**; at a given point in time, a Parcel is a spatial Feature.
  A Parcel may be classified as **having some type of Land Use**.
  There may be other types (**subclasses**) of Parcel, defined in more precise or different ways, such as a Zone.
  A Parcel may have some associated Area. This is currently a variant property and we have yet to determine whether this is equivalent to the area of the Geometry of the Parcel's location (e.g. there may be various values with different accuracy from different sources).
  A Parcel may have some population that is subject to change over time.
  A Parcel may have a number of employed residents that is subject to change over time.
- LandUseClassification: Land Use Classifications provide a means of describing the land cover/use in a standard way. Various classification systems are used to identify types of land use. Currently, we include LBCS, CLUMP, and AAFC.
- The LBCS recognizes different dimensions of Land Use: Activity, Function, Structure, Site, and Ownership Classifications. Each dimension is further defined by a taxonomy of specialized classifications. For each dimension, we introduce an equivalent class name for disambiguation, e.g. to distinguish between the Activity dimension of land use (we refer to this as ActivityClassification) and the notion of an Activity in icity.

    - Activity Classification: An Activity Classification identifies the **activity use** of some Land Parcel.
        - Residential Activities
        - Shopping Activities
        - Industrial Activities
        - ...
    - Function Classification: A Function Classification identifies the **economic function of** some Land Parcel,
    - Structure Classification: A Structure Classification identifies the **type of structure(s) on** some Land Parcel.
    - Site Classification: A Site Classification identifies the **state of the site development on** some Land Parcel (e.g. is it developed or not?)
    - Ownership Classification: An Ownership Classification identifies any **constraints on the use of the land and its ownership for** some Land Parcel.
- CLUMPClassification: Canada Land Use Monitoring Program Classification is a type (subclass) of Land Use classification. CLUMP identifies 15 different types of land use, each with an associated code used in datasets. We have made the design decision that the code need not be unique to a particular land use classification, as a classification from one system may correspond to multiple classifications in CLUMP. CLUMP introduces the following land use classifications:
    - B - Urban built-up area
    - E - Mines, quarries, sand and gravel pits
    - O - Outdoor recreation
    - H - Horticulture
    - G - Orchards and vineyards
    - A - Cropland
    - P - Improved pasture and forage crops
    - K - Unimproved pasture and range land
    - T - Productive woodland
    - U - Non-productive woodland
    - M - Swamp, marsh or bog
    - S - Unproductive land - sand
    - L - Unproductive land - rock
    - 8 - Unmapped areas (technically not a CLUMP classification but it is used in the land use data)
    - Z - Water areas (technically not a CLUMP classification but it is used in the land use data)
- AAFCClassification: Agriculture and Agri-Foods Canada Classification is a type (subclass of) land use classification. The codes are based on the IPCC (International Panel on Climate Change) protocol. We have made the design decision that the code need not be unique to a particular land use classification, as a classification from one system may correspond to multiple classifications in AAFC. AAFC uses the following land use classifications:
    - Unclassified
    - Settlement
    - Roads
    - Water
    - Forest
    - Forest Wetland
    - Trees
    - Treed Wetland
    - Cropland
    - Grassland Managed
    - Grassland Unmanaged
    - Wetland
    - Wetland Shrub
    - Wetland Herb
    - Other land
- TrafficZone: traffic zone is a kind of (subclass of) Parcel. It may be identified with a predefined set of identifiers, corresponding to its centroid node ID.

Object	Property	Value
ParcelPD	subclassOf	change:TimeVaryingConcept
	equivalentClass	change:hasManifestation some Parcel and change:hasManifestation only Parcel
	change:existsAt	exactly 1 time:Interval
	~~spatial_loc:hasLocation~~	~~exactly 1 spatial_loc;SpatialFeature~~
lbcs:Parcel	subClassOf	spatial_loc:Feature
	subclassOf	change:Manifestation
	equivalentClass	change:manifestationOf some ParcelPD and change:manifestationOf only ParcelPD
	change:existsAt	exactly 1 time:TemporalEntity
	hasLandUse	min 1 LandUseClassification
	associatedArea	only om:area
	hasPopulation	only Population
ResidentPopulation	subclassOf	govstat:Population
EmployedPopulation	subclassOf	ResidentPopulation
LBCSClassification	subclassOf	LandUseClassification
ActivityClassification	subclassOf	LBCSClassification
	equivalentClass	lbcs:Activity
FunctionClassification	subclassOf	LBCSClassification
	equivalentClass	lbcs:Function
StructureClassification	subclassOf	LBCSClassification
	equivalentClass	lbcs:Structure
SiteClassification	subclassOf	LBCSClassification
	equivalentClass	lbcs:Site
OwnershipClassification	subclassOf	LBCSClassification
	equivalentClass	lbcs:Ownership
CLUMPClassification	subclassOf	LandUseClassification
	equivalentTo	hasCLUMPCode min 1 xsd:string
AAFCClassification	subclassOf	LandUseClassification
	equivalentTo	hasAAFCCode min 1 xsd:string
Unclassified	subclassOf	AAFCClassification
	equivalentTo	hasAAFCCode value "11"
Settlement	subclassOf	AAFCClassification
	equivalentTo	hasAAFCCode value "21"
Roads	subclassOf	AAFCClassification
	equivalentTo	hasAAFCCode value "25"
Water	subclassOf	AAFCClassification
	equivalentTo	hasAAFCCode value "31"

Forest	subclassOf	AAFCClassification
	equivalentTo	hasAAFCCode value "41"
ForestWetland	subclassOf	AAFCClassification
	equivalentTo	hasAAFCCode value "42"
Trees	subclassOf	AAFCClassification
	equivalentTo	hasAAFCCode value "45"
TreedWetland	subclassOf	AAFCClassification
	equivalentTo	hasAAFCCode value "46"
AAFCCropland	subclassOf	AAFCClassification
	equivalentTo	hasAAFCCode value "51"
GrasslandManaged	subclassOf	AAFCClassification
	equivalentTo	hasAAFCCode value "61"
GrasslandUnmanaged	subclassOf	AAFCClassification
	equivalentTo	hasAAFCCode value "62"
Wetland	subclassOf	AAFCClassification
	equivalentTo	hasAAFCCode value "71"
WetlandShrub	subclassOf	AAFCClassification
	equivalentTo	hasAAFCCode value "73"
WetlandHerb	subclassOf	AAFCClassification
	equivalentTo	hasAAFCCode value "74"
OtherLand	subclassOf	AAFCClassification
	equivalentTo	hasAAFCCode value "91"
UrbanBuiltUp	subclassOf	CLUMPClassification
	equivalentTo	hasCLUMPCode value "B"
MinesQuarriesSandGravelPits	subclassOf	CLUMPClassification
	equivalentTo	hasCLUMPCode value "E"
CLUMPCropland	subclassOf	CLUMPClassification
	equivalentTo	hasCLUMPCode value "A"
CLUMPWater	subclassOf	CLUMPClassification
	equivalentTo	hasCLUMPCode value "Z"
Horticulture	subclassOf	CLUMPClassification
	equivalentTo	hasCLUMPCode value "H"
ImprovedPasture	subclassOf	CLUMPClassification
	equivalentTo	hasCLUMPCode value "P"
NonProductiveWoodland	subclassOf	CLUMPClassification
	equivalentTo	hasCLUMPCode value "U"
OrchardsVineyards	subclassOf	CLUMPClassification
	equivalentTo	hasCLUMPCode value "G"
OutdoorRecreation	subclassOf	CLUMPClassification
	equivalentTo	hasCLUMPCode value "O"
ProductiveWoodland	subclassOf	CLUMPClassification
	equivalentTo	hasCLUMPCode value "T"
SwampMarshBog	subclassOf	CLUMPClassification
	equivalentTo	hasCLUMPCode value "M"
UnimprovedPasture	subclassOf	CLUMPClassification

	equivalentTo	hasCLUMPCode value "K"
Unmapped	subclassOf	CLUMPClassification
	equivalentTo	hasCLUMPCode value "8"
UnproductiveRock	subclassOf	CLUMPClassification
	equivalentTo	hasCLUMPCode value "L"
UnproductiveSand	subclassOf	CLUMPClassification
	equivalentTo	hasCLUMPCode value "S"

Reused Ontologies:

- lbcs: Land Based Classification Standards (LBCS) Ontology[18] presented by (Montenegro, Gomes, Urbano, & Duarte, 2011).
- iCity-Foundation

Future Work:

- In future versions of the ontology, it may be desirable to include an optional relationship for Parcel that identifies its associated organization (e.g. municipal / federal government, transit agency, etc.)
- Future work may extend the population representation to capture various sorts of populations (employed, students, etc)

6.12 Trip Ontology

http://ontology.eil.utoronto.ca/icity/Trip.owl

Namespace: trip

- Trip: A Trip is a kind of **Activity** wherein a Person(s) is transported from one location to another **via some** Mode(s). As with activities, trips may have participants; they may also be described with specialization of the **has participant** property: **hasDriver** and/or **hasPassenger**.
  A Trip **starts at** some **Location** and **ends at** some **Location**.
  A Trip **occurs during** some Interval.
  A Trip **occurs in** some Network(s).
  A Trip **occurs via** some Arc(s).
  A Trip **occurs on** some Transportation Complex. (e.g. a road or a rail)
  A Trip contains some Trip Segments.
  A Trip may incur some cost (monetary or otherwise).
- A Trip Segment describes part of a trip. It may be used, for example, to identify different parts of the Trip by Mode.
  The restrictions on the Mode and possibly Vehicle used will become more complicated as we begin to incorporate restrictions based on a Persons access to a vehicle (age, household).
  A Trip Segment is a specialization of a Trip that is subactivity of some Trip.
  A Trip Segment **occurs during** some Interval.
  A Trip Segment **occurs in** some Network(s).
  A Trip Segment occurs via some Arc(s).
  A Trip occurs on some Transportation Complex.
  A Trip Segment may incur some cost (monetary or otherwise).
- Tour: A sequence of Trips made by one Person.
  A Tour **starts and ends at** the same Location.

Object	Property	Value
Trip	subclassOf	activity:Activity
	startLoc	only spatial_loc:SpatialFeature
	endLoc	only spatial_loc:SpatialFeature

[18] Not available online

	~~during~~	~~exactly 1 time:Interval~~
	accessesNetwork	min 1 transportation:Network
	accessesArc	min 1 transportation:Arc
	occursOn	min 1 transportation:TransportationComplex
	viaMode	min 1 transportation:Mode
	viaVehicle	only Vehicle
	hasDriver	only change:Manifestation
	hasPassenger	only change:Manifestation
TripSegment	subclassOf	Trip
	inverse (hasSubactivity)	min 1 Trip
	viaMode	~~min~~ exactly 1 vehicle:Mode
	viaVehicle	only vehicle:Vehicle
Tour	subClassOf	Trip
	startLoc	startLoc only (inverse (endLoc) Self)

Reused Ontologies:

- iCity-TransportationSystem
- iCity-Vehicle

6.12.1 Trip Costs

http://ontology.eil.utoronto.ca/icity/TripCost.owl

Namespace: tripcost

Different costs are associated with the performance of Trips. These may take the form of direct costs such as those presented in the Travel Cost Ontology, but there may be non-monetary costs associated with travel over different arcs such as pollution and travel time. Trip Costs capture these indirect costs that may vary between individual trips; a trip cost is a property of some instance of travelling.

- A Duration Cost is a Trip Cost.
  A duration cost has an associated cost in terms of duration; e.g. the length of time to perform the trip or trip segment
  A duration cost may have an associated monetary cost (valuation); e.g. the monetary cost applied to the length of time taken to perform the trip or trip cost.
- A Distance is a Trip Cost
  A distance has an associated cost in terms of the distance travelled.
  It may also have an associated monetary cost (valuation)
- An Environmental Cost is a Trip Cost
- A Vehicle Cost is a Trip Cost

Object	Property	Value
TripCost	hasMonetaryCost	only om:MonetaryValue
	tripCostOf	only (trip:Tour or trip:Trip or trip:TripSegment)
DurationCost	subclassOf	TripCost
	hasDurationCost	only time:DurationDescription
DistanceCost	subclassOf	TripCost
	hasDistanceCost	only om:length or om:MonetaryValue

EnvironmentalCost	subclassOf	TripCost
	hasEnvironmentalCost	only CarbonEmissions
VehicleCost	subclassOf	TripCost

Reused Ontologies:

- iCity-Trip

6.13 Urban System Ontology

http://ontology.eil.utoronto.ca/icity/UrbanSystem.owl

Namespace: urbansys

Earlier in this report, we recognized that the urban system covers many different concepts, thus motivating the design of the preceding, so-called generic ontologies. However, it must be recognized that in isolation, these concepts do not effectively capture the urban system. The urban system not only includes these concepts, but relationships between them. For example, the relationship between its population and trips taken and vehicles used. The Urban System Ontology extends all of the previously defined ontologies in order to capture the relationships between them, in the context of the urban system.

- A Person may be a **member of** a Family and/or a Household.
  A Person may work for another Person, or some Organization, or be enrolled at some Educational Institution.
  A Person may **have access to** some **Vehicle**.
  A Person may **have access to** some **Bicycle**.
  A Person has a **Schedule** for a given point (period) in time.
- A Schedule is a plan for some Activity to occur at/over some point in time.
- A Family **has members** who are Persons, and who are related via the has-spouse or has-child properties.
- A Household has one or more Persons as **members**. We do not make any commitment regarding the identity of the Persons, and in fact a Person may belong to more than one Household.
- A Dwelling Unit is **located in some Building** (e.g. House, Apartment,...)
- An Organization must have at least 2 Person(s) as members(s).
- A Firm or a Business Establishment may have a Person as an employee
- An Employee is a type of Person(s).
- Occupation: An Occupation **is performed by** some Person.
  An Occupation has a type (e.g. sales, skilled trades)
- A Building may be located on some Parcel of land (this is an invariant property of any building).
  A Building **has an owner**, which may be a Persons or some Organization.
  A Building **has occupants**, which may or may not be the same Persons or Firm who own it.
  A Building may **provide** some **Parking**.
- A Building Unit may be **occupied by** some Persons or Organization.
  A Building Unit may be **provide some Parking**.
- A Vehicle may be **occupied by** at least one Person, and some cargo.
  A Vehicle is **owned by** some Person(s) or Firm.
  A Vehicle has some associated **Mode**.
- Occupant: An occupant is a Person who is occupying a Vehicle during transit.
  An Occupant may be a Driver or a Passenger
- Cargo: A Cargo is some Thing that is not a Person and is occupying a Vehicle during transit.
- An *entire* Arc is accessible by a single set of Mode(s).
- A Road Segment is accessed by some Arc(s) with modes that are not water, air, or rail.
- A Parking Area has some **owner**.
  A Parking Area may be **occupied by** some Vehicle (however, it might also be occupied by some debris or activities such as construction).

- A Parking Policy may **apply to** a specific group of Persons or Organizations. A Parking Policy may have a **vehicle type restriction.**
- A TransitSystem may be **owned by** some Organization.
- A Route is **executed by** various Vehicles at different points in time.
- A Trip is **made by** a Person to **facilitate participation in some Activity**.

Object	Property	Value
person:Person	memberOf	min 1 household:Family
	memberOf	min 0 household:Household
	schema:worksFor	some (person:Person or org:Organization)
	hasAccess	some (vehicle:Vehicle or Bicycle)
	hasSchedule	some Schedule
Schedule	hasActivity	only activity:Activity
	scheduledFor	exactly 1 time:Interval
household:Family	hasMember	only person:Person
household:Household	hasMember	min 1 (household:Family or person:Person)
household:DwellingUnitPD	locatedIn	some building:Building
org:Organization	org:hasOrgMember	min 2 person:Person
org:Firm	hasEmployee	only person:Person
org:BusinessEstablishment	hasEmployee	only person:Person
org:Employee	equivalentClass	person:Person and employedBy some (tove:Organization or person:Person)
Occupation	performedBy	some person:Person
	hasOccupationType	only OccupationType
building:BuildingPD	locatedOn	only landuse:Parcel
building:Building	hasOwner	min 1 (person:Person or org:Organization)
	hasOccupant	some person:Person or org:Organization or org:BusinessEstablishment
	hasParking	only parking:ParkingArea
vehicle:Vehicle	occupiedBy	only (Occupant or Cargo)
	hasOwner	only (person:Person or org:Organization)
	hasMode	only transportation:Mode
Occupant	equivalentClass	person:Person and occupies some vehicle:Vehicle
Cargo	equivalentClass	not(person:Person) and occupies some vehicle:Vehicle
transit:TransitSystem	hasOwner	only org:Organization
transit:Route	executedBy	only vehicle:Vehicle
trip:Trip	subClassOf	activity:Activity
	performedBy	some person:Person
	associatedWith	only activity:Activity

Future work:

- May be useful to add a generalized 'hasPass' relationship to capture various possible passes a person may have (transit and otherwise)

7 Applications

Beyond providing formalizing a vocabulary for an integrated, iCity knowledge base, the ontology may be employed more directly support applications for urban informatics. Three such examples are currently being explored: (1) ontology support for the IT-SoS framework, including the development of a semantic trip planner; (2) ontology support for survey design and results storage; and (3) ontology-based support for simulation result question-answering. In this section we focus on the IT-SoS application and describe the progress made thus far. More detail will be added as each application is explored further.

7.1 iCity Ontology for the IT-SoS Framework

The IT-SoS framework proposed by [ref Elshenawy] enables a solution for the design of transportation applications capable of dynamically executing services based on a given context. Using the framework, services can easily be added and integrated as part of applications as appropriate.
[more detail/diagram]
Ontologies play a key role in this framework. Here, we focus on the interface between the iCity ontology and the rest of the system. This interface must be well-understood and clearly defined in order to implement this framework in the context of the iCity project. In the following sections, we outline the functional requirements and describe the design of an interface to satisfy these requirements.

7.1.1 Ontology Interface: Functional Requirements

More detail on the complete framework can be found in [ref]. The role of the ontology and its interface with the rest of the system is specified with the following requirements (revised from the requirements identified in [ref thesis]):

- Ontology representation of services (WFS)
  - To support discovery
  - To support composition
- Ontology representation of applications
  - To enable (context-based) composition

The above requirements are formalized and elaborated with the following use cases:

1. A user formalizes the relevant semantics of a new WFS service for the ArcGIS server, to be part of the IT-SoS framework.
2. Given a process definition and some contextual information, the system identifies the information requirements required to perform it.
3. Given an information requirement, identify which services are capable of providing the required information.

Use Case 1	A user formalizes the relevant semantics of a new WFS service for the ArcGIS server.

Goal in Context	Users create new WFS services for the ArcGIS server, to be incorporated into the IT-SoS framework.	
Preconditions	WFS service designed correctly.	
Success End	WFS service created, formalized with metadata.	
Failure End	WFS service not created and/or metadata not captured	
Primary & Secondary Actors	Primary: user	
Trigger	New WFS service created.	
Description	**Step**	**Action**
	1.	User creates new WFS service.
	2.	User defines additional required metadata for service.
	3.	User submits the service to IT-SoS.
	4.	Service description formalized in the language of the ontology.
Extensions	**Step**	**Branching Actions**
Variations	**Step**	**Branching Actions**
	2a.	No sources are capable of supplying all of the required data 2a1. System or user notified query is unknown...
Related Information		
Priority	High	
Performance	TBD	
Frequency	High	
Open Issues	What metadata is required?	

Use Case 2	Given a process definition and some contextual information, the system identifies the information requirements required to perform it.
Goal in Context	In order to perform automatic and dynamic composition of applications, the system must be able to determine what information is required in a particular context.

Preconditions	Application process correctly formalized. Contextual information available	
Success End	All and only the correct information requirements are identified.	
Failure End	Information requirements not (fully or exclusively) identified.	
Primary & Secondary Actors	Primary: application composer	
Trigger	Application execution.	
Description	**Step**	**Action**
	1.	Application process selected.
	2.	Contextual information input.
	3.	Information requirements to support application process generated.
Extensions	**Step**	**Branching Actions**
Variations	**Step**	**Branching Actions**
Related Information		
Priority	High	
Performance	TBD	
Frequency	High	
Open Issues		

Use Case 3	Given an information requirement, the system identifies which services are capable of providing the required information.
Goal in Context	In order to perform automatic and dynamic composition of applications, the system must be able to determine which services are capable of satisfying which information requirements.
Preconditions	WFS services correctly formalized. Application process correctly formalized.
Success End	All and only the required services are identified.
Failure End	Required services not (fully or exclusively) identified.

Primary & Secondary Actors	Primary: application composer	
Trigger	Application execution.	
Description	**Step**	**Action**
	1.	User provides a set of information requirements.
	2.	WFS services capable of satisfying the information requirements are identified.
Extensions	**Step**	**Branching Actions**
Variations	**Step**	**Branching Actions**
Related Information		
Priority	High	
Performance	TBD	
Frequency	High	
Open Issues		

7.1.2 System Design

Illustrated in Figure 9, the creation of ITS applications using the IT-SoS framework is supported within the iCity project with three key components: the WFS creator, which supports the definition of individual services; the Application creator, which supports the definition of applications as processes; and the Application engine, which supports the automated composition and execution of services.

The WFS Creator (illustrated in Figure 10) supports the creation of a WFS service from some existing data source. The WFS is created and published to the ArcGIS server. Additional metadata is also collected during the creation process in order to achieve a complete description of the service (e.g. the equipment it uses, the information it provides). This description is mapped into a formal representation using terminology defined in the iCity ontology, and stored in a triplestore which may be accessed as required by other components.

The Application Creator component supports the definition of ITS applications – in particular the context-specific decomposition of the underlying process (including accounting for possible dependencies between sub-processes). As with the creation of WFS services, each application definition shall be mapped into an ontology-based representation in the IT-SoS triplestore.

The iCity ITS Application Engine (illustrated in Figure 11) is responsible for building and executing ITS applications on-the-fly, according to the definitions specified by the ITS Application Creator.
In the context of the iCity project, each application shall be accessible (buildable and executable) via the iCity dashboard. Based on the selected application and additional contextual information (supplied by the dashboard), the engine queries the triplestore for the appropriate composition in order to determine how the application shall be built. The resulting composition then serves as input for the execution engine, which combines the WGS services from the ArcGIS server in accordance with the prescribed composition in order to execute the application. Since all of the WFS services are represented using the iCity ontology, information between services may be combined easily, using the ontology as the interlingua.

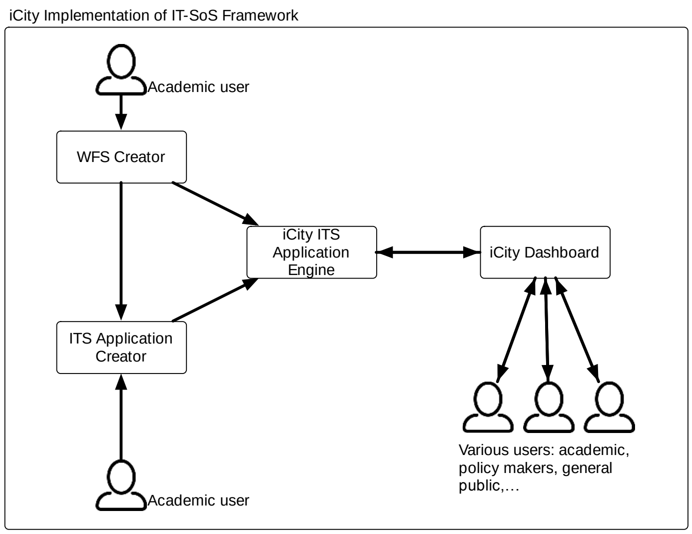


Figure 9: Ontology-focused view of the iCity implementation of the IT-SoS framework.

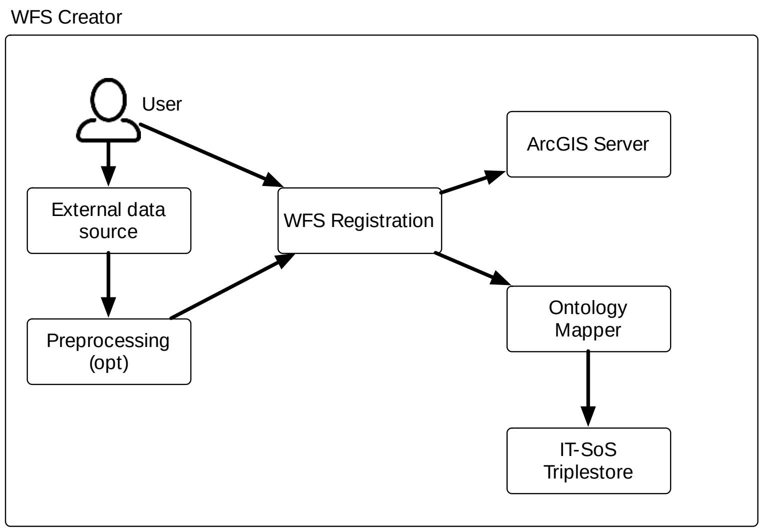


Figure 10: Illustration of the key components within the WFS Creator.

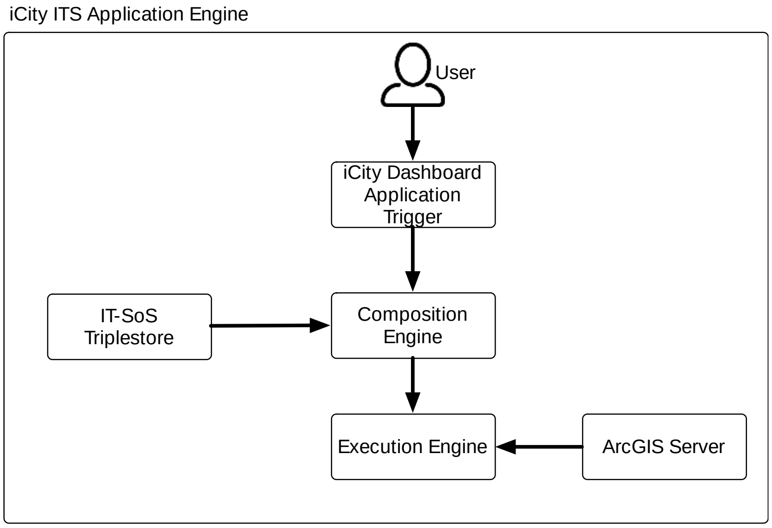


Figure 11: Illustration of the key components within the ITS Application Engine.

7.1.3 Ontology Design: Required Extensions

The iCity ontology must be extended in order to capture WFS and their possible compositions. This will be approached by formalizing some sample application definitions and compositions in order to guide the metadata that will be required to support the system's functionality.

7.1.4 Next Steps

The next steps toward implementation of this application will be to clearly define the required, ontology-based representation of applications and services within the system. This will be incorporated into a workflow that will become part of the system's processes to add applications and services. In this way, the ontology interface to the IT-SoS system will be baked-in to the creation process, requiring no additional overhead on the part of the user. The system shall be implemented to take advantage of the ontology-based representation in order to support the automated, intelligent composition of applications as envisioned by the IT-SoS framework.

7.2 Analysis of TTC Data for Bus Bridging Study

7.3 Organization of Travel Model Data

7.4 iCity Ontology for Urban Simulation Results

Urban System Simulation Ontology
UrbanSystemSimulation.owl

Ontologies present the opportunity to concurrently address multiple challenges for urban modelling and simulation, such as:

1. the incomparability between models and results,
2. the need for bespoke query design, and
3. the opaqueness/complexity of models and results.

In this application, we focus on the use of the ontology to formalize the simulation results. Future work should extend this to focus on the models and the simulation runs.

The result of an urban system simulation is essentially an instance of some part(s) of the urban system and can be formalized by the urban system ontology. In addition, we need a way to distinguish such instances from real-world data. To accomplish this, we extend the Urban System Ontology with an ontology for simulations: the Urban System Simulation Ontology. The following concepts are required for the Simulation extension:

- Simulation: A Simulation is an execution of some model system. It has some input and **output** data, defined by some instances of the UrbanSystemOntologyClass.
  A Simulation has a **run date**.
-

Object	Property	Value
Simulation	hasSimulationOutput	some UrbanSystemOntologyThing
	hasRunDate	exactly 1 xsd:dateTime

Future work: Additional aspects of the simulation (e.g. inputs, run dates, models used) are relevant, and should be defined in future work. They are discussed in Section 8.2.2.

8 Future Work

The iCity Ontology, presented in the previous section, has been classified with the Hermit reasoner in Protegé 5.1[19] and shown to be consistent. An initial, informal evaluation has been performed through a review of its contents with iCity project members serving as domain experts. Future iterations shall be informed by and evaluated against a more precisely defined series of competency questions to be elicited from the iCity project team.

Future iterations of the iCity Ontology will develop a deeper semantics for the concepts identified here, in addition to an expansion of scope. This will be dictated largely by use cases identified by the various project groups, which will not only determine additional requirements for representation, but potential applications for additional functionality that may be supported by the ontology.

8.1 Extensions to the Urban System Ontology

In developing a richer semantics for the iCity concepts, we will also look to identify more detailed connections between them. This will serve to facilitate shareability between the various projects and domains within iCity. Consider for example, the identification of relationship between common property types, such as hasId, memberOf. While there is likely a shared semantics between these relations in, for example the Person/Family and the Organization ontology, in this initial release, we opt to maintain a distinction between these relations (through specialized names, e.g. personId). Future work should, if required, investigate and make explicit exactly what the relationship is.
In a similar vein, future work will also look to integration of the iCity ontology with other existing vocabularies, which may provide opportunities to improve its shareability. For example, in the design of the iCity ontology we identified some vocabularies that were not directly reusable, (specified as XML schemas, for example), however based on their applications, it

[19] http://protege.stanford.edu/

might be advantageous to incorporate the representations in some way. For example, GTFS [20], the format used by Google for travel information.

8.2 Extensions for iCity Applications

The first release of the iCity ontology is designed to capture the urban system. However, we anticipate additional concepts will be required for each iCity project to capture the nature of the data within a given application. Varying definitions of concepts within the urban system should be captured as part of the appropriate ontology (for example, multiple definitions of a Household should be represented by different definitions of Household in the Household ontology), on the other hand the iCity projects also introduce other concepts that are beyond the domain of the urban system, and more related to the applications themselves. For example, a simulation may produce output that captures information about an urban system, but we must also represent that this information is the result of a particular model being applied to some data to explain how it was generated and why it is of interest. We divide the iCity projects into 4 categories based on the nature of the applications: Data Collection, Simulation, Analysis, and Visualization. In the following subsections, we consider the classes and properties for each extension. The resulting structure for this future state of the iCity Ontology is illustrated inFigure 12.

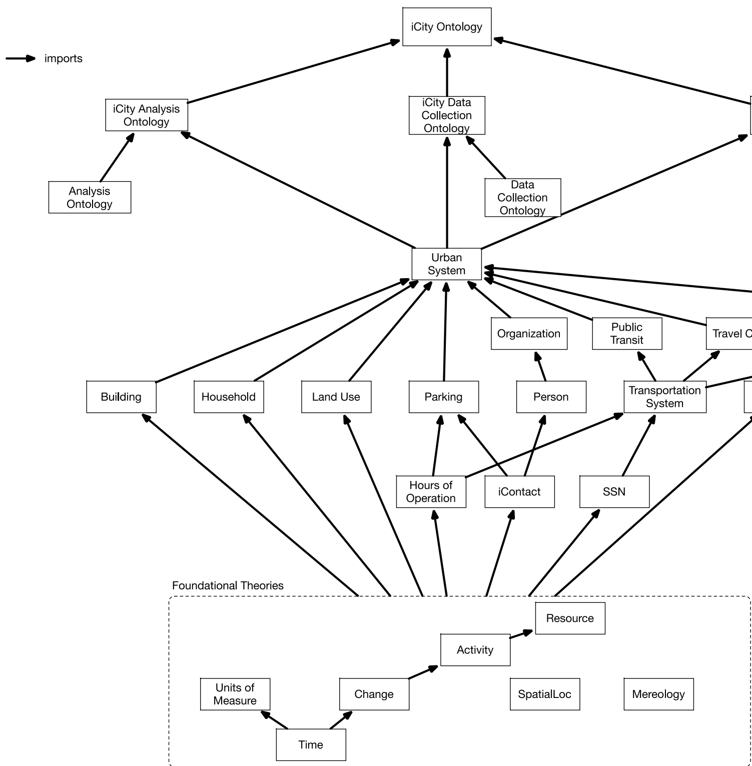


Figure 12: iCity Ontology Structure

In identifying these concepts, a key question is: "What question(s) is the project/application trying to answer?"

[20] https://developers.google.com/transit/gtfs/

Note that it is unclear whether or to what degree there may be some overlap between the requirements for Analysis and Simulation in that they both require some aspect of experiment management. This report concludes with some preliminary notes on the requirements for each category of application in the following sections.

8.2.1 Data Collection

Related projects: 1.2, 1.3, 2.1, 2.2, 2.3

To completely capture collected data requires representation of its origin: what was the means of collection? When was it collected? How may the data be accessed? It requires the representation of concepts *about* the data collection itself. The following additional concepts may be required for the data collection extension:

- Data Entity: A Data Entity refers to some instance that is defined within the urban system, according to some source.
  A Data Collection is a type of (**subclass of**) Data Entity.
  A Data Collection **contains** one or many Data Entities.
  A Data Entity **is generated by** some Collection Activity.
  A Data Entity may be found **at some Location**.
- Data Entity: A Data Entity is any instance **contained in** some Dataset.
- Collection Activity: A Collection Activity indicates the origin of the data; i.e. how was it collected?
  A Collection Activity **starts** and **ends** at some Time
  There are different types (**subclasses**) of Collection Activity: Survey Activity, Sensor Activity, Data Fusion Activity, Simulation Activity, etcetera.
  A Collection Activity may be found **at some Location** (e.g. location of the sensor or survey, could be physical or virtual).
- Data Fusion: A Data Entity may be the result of the Fusion of two or more Data Collections.
  Data Fusion **is informed by** at least 2 Collection Activities.
- Data Collection Agent: The agent responsible for some Collection Activity.
  A Collection Activity may be **associated with** some Data Collection Agent.
  A Data Entity may be **attributed to** some Data Collection Agent.

8.2.2 Simulation of Urban Systems

Related projects: 2.2, 2.3, 2.4

Capturing the simulation activities that occur within the iCity project, at this stage, appears to be very much an effort of experiment management. We need to be able to represent the simulation runs that are performed -- but also, more specifically the model(s) that was used, as well as the results that were obtained. The following additional concepts may be required for the Simulation extension:

- Simulation: A Simulation is an execution of some Model System.
  A Simulation **executes** some Model System.
  A Simulation has some **input** and **output** Dataset(s)
  A Simulation has an **initial** State, **sequence** of States, and **final** State.
  A Simulation has a **run date** and **duration**.

- State: A State is **comprised of** some instantiation of (part of) the urban system, at some specified point in time.
- Model System: A Model System is some configuration of model(s) that has been designed for simulation.
  A Model System **contains** some Model(s)
  A Model System may contain rules for how the Model(s) interact. (sequentially, in parallel, etcetera).
- Model: A Model is a means of advancing some current state within a Simulation.
  A Model **applies to** some classes in the domain.
  There are different types (**subclasses**) of Models, identified based on their perspective: State-oriented Model, Event-oriented Model, Activity-oriented Model, PD-oriented Model.
  A Model **has** some **Parameter**(s).
  A Model may **execute in parallel with** some other Model(s).
  A Model may **execute directly after** some other Model(s).
- State-oriented Model. There are different types (**subclasses**) of State-oriented Models that can be defined, according to the application.
  A State-oriented Model has some State Space
  A State-oriented Model has some Event Set
  A State-oriented Model has some Time Set
  A State-oriented Model has some Transition Function to transition between states.
  A State-oriented Model has some Clock Function to advance "time".
  A State-oriented Model has some Initial State.

8.2.3 Analysis of Urban Systems

Related projects: Project 1.2, 2.1, 2.2, 2.3, 2.4

Similar to the previous section, capturing the various analysis applications may be seen as a sort of experiment management. We must capture the concepts of analysis input, output, as well as the analysis itself: in other words, how is the output determined from the input? The following concepts may be required for the Analysis extension:

- Analysis: A set of rules or criteria applied to some Analysis Input to obtain some Analysis Output.
  An Analysis may take only certain class(es) of instances as Input.
  An Analysis will output only certain class(es) of instances as Output.
- Analysis Input: An Analysis Input is input for some Analysis.
- Analysis Output: An Analysis Output is output from some Analysis.

8.2.4 Visualization of Urban Systems

Related projects: All of Theme 3

The concepts defined in the iCity ontology (and the data they define) shall be interpreted for visual renderings; to-date no additional requirements have been identified.

9 Extra-logical Design Practices

Here, we summarize and explain the design practices that were adopted in the creation of the ontologies. These practices do not pertain to the semantic definitions, but rather are adopted to address pragmatic concerns regarding the organization and maintenance of the ontologies.

- Organizational terms
- For reuse (full import) of existing, external ontologies, e.g. owl-time. In order to create the required groupings under organizational subclasses, it is easiest to merge the imported ontology into the iCity container (e.g. icity/Time/). This allows for the addition of organizational subclass assertions (e.g. TemporalEntity subclassOf TimeOntologyThing) and also ensures that the appropriate version is captured/reused as a snapshot. This prevents any issues should versioning IRIs not be used by the ontology's author.

Acknowledgements

This project is supported by the Ontario Ministry of Research and Innovation through the ORF-RE program.

Bibliography

Bittner, T., & Donnelly, M. (2005). Computational ontologies of parthood, componenthood, and containment. *Proceedings of the 19th international joint conference on Artificial intelligence (IJCAI).* Morgan Kaufmann Publishers Inc.

Fadel, F. G., Fox, M. S., & Gruninger, M. (1994). A Generic Enterprise Resource Ontology. *Third Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises* (pp. 117-128). IEEE.

Fox, M. S., Barbuceanu, M., Gruninger, M., & Lin, J. (1998). An Organization Ontology for Enterprise Modelling. In K. C. M. Prietula, *Simulating Organizations: Computational Models of Institutions and Groups* (pp. 131-152). Menlo Park CA: AAAI/MIT Press.

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

Grau, B. C., Horrocks, I., Motik, B., Parsia, B., Patel-Schneider, P., & Sattler, U. (2008). OWL 2: The next step for OWL. *Web Semantics: Science, Services and Agents on the World Wide Web*, 309-322.

Hobbs, J. R., & Pan, F. (2004). An ontology of time for the semantic web. *ACM Transactions on Asian Language Information Processing (TALIP)*, 66-85.

Krieger, H.-U. (2008). Where temporal description logics fail: Representing temporally-changing relationships. *Annual Conference on Artificial Intelligence* (pp. 249-257). Springer Berlin Heidelberg.

Lorenz, B., Ohlbach, H. J., & Yang, L. (2005). *Ontology of transportation networks.*

Montenegro, N., Gomes, J., Urbano, P., & Duarte, J. (2011). An OWL2 Land Use Ontology: LBCS. *International Conference on Computational Science and its Applications* (pp. 185-198). Springer Berlin Heidelberg.

Welty, C., Fikes, R., & Makarios, S. (2006). A reusable ontology for fluents in OWL. *Formal Ontology in Information Systems (FOIS)*, (pp. 226-236).