GenCLOn: An ontology for city logistics

Nilesh Anand*, Mengchang Yang, J.H.R. van Duin, Lori Tavasszy

Delft University of Technology, Jaffalaan 5, 2628BX Delft, The Netherlands

1. Introduction and motivation

With the development of city logistics discipline, it has been widely accepted that a key characteristic of this domain is high number of stakeholders and the heterogeneity of their needs. This forces a very thin segmentation of the proposed services, often going below the minimum scale for economic feasibility (Macário, Galelo, & Martins, 2008). Succinctly, stakeholders involved in the city logistics domain have their own interests (see Table 1) and thus act autonomously without any centralized control. For example, shopkeepers order small but frequent deliveries to reduce their inventory cost, which creates more nuisances from more freight vehicles. Administrators, on the other hand, impose regulations such as weight restrictions to mitigate the nuisance from freight vehicles at the sacrifice of carriers’ profit. This situation makes urban freight transport a distributed decision making system and in turn lead to the visible problems in urban freight transport (e.g. poor economic and environmental sustainability). Accordingly, there is a strong need for a systematic and analytical approach to grasp decision making among different stakeholders in order to understand urban freight transportation movements (Anand, Duin, Quak, & Tavasszy, 2011).

Research in city logistics domain is performed to facilitate an efficient urban freight delivery system and economic growth. City logistics modeling works as forecasting tool to gain insight about current (and future) freight transportation, commodity flow, infrastructure and information needs. City logistics modeling helps to create a knowledge base about truck flow, commodity flow, behavioral insight of different stakeholders involved etc. This knowledge base is used to understand and predict city logistics trends and problems in attempt to invent policy measures and initiatives aiming at proper functioning of city logistics.

Problems related to city logistics are visible in city traffic, but its roots are connected to decision making of different actors it involves. To model a city logistics domain is important but not trivial task since in order to get an efficient communications many stakeholders should interoperate with each other sharing a common knowledge with different terminologies and types of decision they are making. From the semantics point of view, a common language (at least a glossary) is needed in order to have coordination among users, systems and communications. Thus, it is important to codify semantics in an accessible way so that it is easy for users to interpret notifications and for expert users to communicate among each other on relevant topics.

Among several modeling platforms available for city logistics (henceforth CL) domain, agent based modeling is the latest addition. Agent-based model (ABM) can be a valuable tool among CL researchers for its strong capability of capturing the dynamic behavior of individual stakeholders and their interconnections (Getchell, 2008). Initial attempts for ABMs in CL domain are found in works of (Donnelly, 2009; Kolck, 2010; Tamagawa, Taniguchi, &
These models are developed to evaluate the potential impact of different policy measures in CL domain. ABM is a powerful tool, nevertheless the human effort required to develop an ABM tends to be huge, especially when multiple stakeholders have to be covered (Tamagawa et al., 2010). An ABM has to be constructed upon a knowledge base (Le Ber & Chouvet, 1999) which abstracts a specific domain into a world purely composed of agents and their relationships. However, today, this kind of knowledge bases are still built with little sharing or reuse – almost each one starts from a blank slate. ABM developers have to go to (abundant) literatures and data, seek the terms of interest, sort the complex, and usually implicit relations among terms, and code the agents together with their properties (i.e. behavioral rules and communication protocols) merely on the basis of what they subjectively learn without a shared reference. This situation directly leads to a poor reusability of models that ultimately incurs repetitive work and extra developing time. For instance, a developer wants to add a new agent to her model and is informed that there exist some models containing this agent. Unfortunately, after checking the models of interest, she finds that agents in these models are coded in a different way which is non-compatible/inconsistent with her model. Helplessly, she has to spend significant amount of time on coding the agent herself. In this situation, it is imaginable how easier the life would be if there is a shared knowledge base using which ABM developers can standardize their models and consequently reach interoperability.

Fortunately, the introduction of the CL ontology can change, or at least improve, this situation. The term ‘ontology’ was coined by Aristotle as a branch of metaphysics, and is the study of ‘the nature and structure of reality’. According to (Gruber, 2008), the traditional goal of ontological inquiry is to ‘divide the world at its joints, to discover those fundamental categories, or kinds, into which the world’s objects naturally fall’. Since the mid-1970s, researchers have been recognizing the great value of ontology for capturing knowledge, which is regarded as the key to building large and powerful Artificial Intelligence (AI) system. In the 1980s, the AI community began to use the term ontology to refer to both a theory of a modeled world and a component of knowledge systems. In the early 1990s, (Gruber, 1995) further defined ontology into a technical term in computer science as ‘a specification of a conceptualization’. That is ‘an ontology is a description (like a formal specification of a program) of the concepts and relationships that can exist for an agent or a community of agents’. In other words, ontology stipulates the potential agents and their rules of interaction in a given domain. Following this discussion, it’s conceivable that a CL ontology can abstract the CL domain into multiple concepts/classes. Some of these classes can directly be adopted as agents in the models, and their potential interactions have already been formally clarified in the ontology. Hence, CL-ABM developers can have a shared and standardized template that specifies the data structure and initial data used within their models. Rather than building from scratch, they are able to assemble their knowledge bases with components drawn from the ontology. Furthermore, specific agents in an existing model can be directly reused in other models as long as their developers follow the same ontology (Keirstead & van Dam, 2010). All of these merits should greatly decrease development time while improving the robustness and reliability of the resulting knowledge bases. An intuitive problem forming mechanism is given in Fig. 1.

The remainder of the paper is organized as follows. In Section 2 we present literature review. In the Section 3, we conceptualize the CL domain and reveal the scope of CL ontology. In the next section – Section 4 – we start defining CL ontology with its main classes/concepts that is followed by CL ontology axioms and relationships in Section 5. Section 6 describes the validation of CL ontology using data driven approach. Section 7 illustrates applications of CL ontology and in Section 8 conclusions is presented.

### Table 1

<table>
<thead>
<tr>
<th>Stakeholder</th>
<th>Interest</th>
</tr>
</thead>
<tbody>
<tr>
<td>Residents</td>
<td>Product and Services</td>
</tr>
<tr>
<td></td>
<td>Negative environmental impact</td>
</tr>
<tr>
<td>Retailers</td>
<td>Competitiveness and profitability</td>
</tr>
<tr>
<td>Authorities and Public Service (Administrator)</td>
<td>Accessibility</td>
</tr>
<tr>
<td></td>
<td>Governance and legislation</td>
</tr>
<tr>
<td></td>
<td>Negative environmental impact</td>
</tr>
<tr>
<td>Supplier</td>
<td>Market growth</td>
</tr>
<tr>
<td></td>
<td>Profitability</td>
</tr>
<tr>
<td>Carrier</td>
<td>Congestion</td>
</tr>
<tr>
<td></td>
<td>Cost effectiveness</td>
</tr>
</tbody>
</table>

2. Literature review

Although so far there is no explicit contribution to ontology dedicated to CL is available in literature, ontology is not something new in the domain of logistics/supply chain management – which has a close relationship with CL. Several initiatives have been taken by researchers to formalize the domain knowledge with ontology. Lian, Park, and Kwon (2007) proposed an ontology that as a logistics situation model which can help to specify a situation of a product and attach events that ‘triggered’ by this situation. Different from the company-oriented efforts, the main application of this ontology is to help construct scenario of logistics process, which seems more academic. The ‘trigger’ concept here will probably make great sense for simulating the behavior of agents in the CL domain. In another attempt, Leukel and Kirn (2008) made a creative trail to build an logistics ontology with the knowledge contained in the famous SCOR (i.e. Supply Chain Operation Reference) model. Due to the partial focus of SCOR model on logistics process and information flow, this ontology has a relatively poor representativeness of the physical/material aspect (e.g. goods, infrastructure, vehicles etc.) of the logistics domain. However, it is still a powerful reference for building the abstract parts of the CL ontology.

Hoxha, Scheuermann, and Bloehdorn (2010) also contributed an ontology for the semantic representation of the logistics domain, which offers solutions to the integration challenges among heterogeneous data and interoperability of logistics services from different providers. In this attempt, the authors divide the logistics domain into five top classes – namely actors, logistics service, logistics process, logistics resource and logistics KPI, and each class has been further specified with multiple sub-structures. Similarly, Zhang and Tian (2010) designed a logistics domain ontology model to represent the relations among logistics domain concepts to facilitate integration of inter-enterprise logistics information system. This ontology contains 12 top classes – namely cargo, organization, supplier, customer, carrier, transport service,
constraint (e.g. price, time limit), transport service standard (i.e., demand about transport service), transport mode, vehicle, traffic line and information between actors. Compared to the previous case, classification asserted by Hoxha et al. (2010) tends to be more systematic as well as hierarchical, and thus possess considerable referential value. Additionally, they also provide constructive insight in facilitating semantic web-based logistics service request and execution based on their ontology. Other attempts with similar aim are found in work of Liou and Chang (2008), Smirnov and Chandra (2000), Fayez, Rabelo, and Mallaghaseemi (2005), Ye, Yang, Jiang, and Tong (2007), Huang and Lin (2010), Yu-Liang (2010), Xu, Wijesooriya, Wang, and Beydoun (2011). Following conclusions can be drawn after this brief review. Firstly, these previous efforts do lay a good basis for building the micro part (i.e. demand–supply pattern among private actors) of the CL ontology. Secondly, due to the different focus, the macro part (i.e. political, social and environmental issues) exclusively covered by CL has to be constructed from almost a blank slate.

3. Generalization and conceptualization of the city logistics domain

3.1. City logistics domain

Urban freight transport is faced with great challenge nowadays. At first, the overall ever-increasing wealth as well as population1 in urban areas is stimulating the consumption level of citizens. With more money governable, people tend to buy more, which directly lead to the boom of freight flow into the cities. Secondly, not only the amount, but also the variety of products consumed in urban areas has increased significantly. For one type of goods, there can be a number of suppliers for choice, and each of them has own distribution channel. As a result, it becomes difficult to achieve consolidation. Last but not least, the popularity of modern logistics concept such as “Just-In-Time” (JIT) and “smart retailing” stimulate small but frequent deliveries. The direct consequence of these inevitable trends is a less sustainable city haunted by congestion, pollution, inefficient use of resources, which affects the economy as well as livelihood of city inhabitants.

To facilitate an efficient urban freight delivery system and in turn economic growth, city logistics, a specialized logistic domain was brought out by initiators including K. W. Ogden and Eiichi Taniiguchi and is getting enriched continuously. Although the very first reference to this term cannot be ascertained, its history can probably be traced back to the 1970s, a period dedicated to urban freight transportation issues and yielded traffic regulation to avoid the presence of heavy vehicles in cities and thus limit the impact of the freight transport on automobile movements. Very little activity took place in the 1980s since the problems were still under the control of the work before. The increased traffic-related problems and the associated public pressure have revived the interest in CL domain from 1990 onwards and have resulted in traffic surveys and data collection activities, research projects and experimental deployments (Crainic, Ricciardi, & Storchi, 2009). The foundation of the ‘Institute for City Logistics’ (ICL) in Kyoto, Japan in 1999 could be considered as a milestone symbolizing the beginning of a ‘golden age’ of CL.

3.2. Scope of city logistics ontology

Institute of City Logistics defined CL as ‘the process for totally optimizing the logistics and transport activities by private companies in urban areas while considering the traffic environment, the traffic congestion and energy consumption within the framework of a market economy’. This definition suggests that the focus of CL lies in the company-driven logistics activities triggered by business between companies or even between companies and individual consumers. Possible patterns in this case mainly include:

- traditional direct B2B delivery from wholesalers/end goods producers/importers to independent retailers or other business parties located in the urban area;
- dedicated delivery organized by retail chains from DCs to outlets;
- B2C delivery from retailers (outside or inside the same urban area) to urban inhabitants (i.e. home delivery) performed by freight vehicle;
- and logistics activities involved that will influence the urban area such as (un)loading and (part of) return trips, etc.

Correspondingly, some patterns become less important and can even be left out from the scope under the definition. For instance, scattered shopping trips generated by households with private cars are of less importance for that proactive participation from business parties are missing in this pattern. The congestion caused by private cars, however, is still needed to be taken into account since it will influence the performance of urban freight transport anyway. Mail (parcel) express consigned by individuals (including C2C business) will not be taken into account, and the reason is twofold. At first, a demand–supply relation with companies is missing in this pattern. Despite that 3PL parties are also kind of ‘companies’, they are not the ones that really trigger the logistics activities. Secondly, just as the car traffic, the negative impact caused by C2C deliveries to the urban sustainability is low due to both the efficiently organized trips and the vehicles used that tend to be smaller as well as cleaner than the trucks used for B2B business (Ports, 2005).

Following the definition, we can further scope out the upriver actors (e.g. upriver DCs, suppliers of raw materials) in the supply chains that don’t have a direct demand–supply relationship with the receivers (i.e. retailers, inhabitants) situated in the urban areas. Fig. 2 illustrates the domain of interest determined so far for developing CL ontology. To sum up, deliveries from end depots to urban premises/homes by freight vehicles, and deliveries from retail premises to home by freight vehicles, and their return trip along with auxiliary logistics activities that will influence CL performance are of interests in the following work. A series of assumptions will be given in the next section (along with ontology description) to further refine the scope based on the conclusion derived above.

4. Formal specification of generic city logistics ontology (GenCLOn)

4.1. Hierarchy of GenCLOn

Several possible approaches exist in developing a class hierarchy (Uschold & Gruninger, 1996). A top-down development process

4 According to Tan vom Hove (Hove, 2004), the world’s urban population will grow from 2.86 billion in 2000 to 4.98 billion by 2030.
starts with the definition of the most general concepts in the domain and subsequent specialization of the concepts. A bottom-up development process starts with the definition of the most specific classes, the leaves of the hierarchy, with subsequent grouping of these classes into more general concept. A middle-out development process is a combination of the both the methods mentioned above. It defines the more salient concepts first and then generalizes and specializes them appropriately.

None of these three methods is inherently better than any of others. The approach to take depends strongly on the personal view of the domain. A top-down approach results in better control of the level of detail, however demands very systematic top-down understanding of the domain, which is not the best choice for highly dynamic and complicated systems such as CL. A bottom-up approach, instead, facilitates a quick start without imposing high level understanding on the users, however tends to involve over-high level of detail that will easily arouse problems like incremental overall effort, difficulty in spotting commonality between related concepts and risk of inconsistence. Contrarily, both, Uscold and Gruninger (1996) and Noy and McGuinness (2001) believe that the middle-out approach makes it easier to relate terms in different areas more precisely and thus avoid potential for re-work since the concepts ‘in the middle’ tend to be the most descriptive concepts in the domain (Rosch, 1999). Considering its empirical compliments together with the complexity, extensiveness and dynamicity of CL system, the middle-out approach is thus adopted for the following hierarchy construction. Also, due to its non-stipulated characteristic for various CL scenarios, we will, henceforth, call it Generic city logistics ontology – in short GenCLOn.

4.2. GenCLOn classes

In this section the hierarchical structure of the proposed ontology will be mapped out following the middle-out approach. We will at first create the most important classes and then expand from it.

4.2.1. Stakeholders

The middle-out approach advocates determining the most ‘salient’ terms at first and expanding the structure from it. The question arises that ‘which term is the core of CL ontology?’ From the research goal and competency questions, it is clear that this ontology should provide sufficient understanding into the ‘interests and decision making process of heterogeneous stakeholders involved in the CL domain’. Thus it is rational to assert the core status to stakeholders in this ontology (Browne, Woodburn, & Allen, 2005; Taniguchi & Tamagawa, 2005).

The stakeholder can be further classified into two general categories as Private and Public stakeholder which is given naming convention as Private_actor and Public_authority in GenCLOn. Conventionally, private actor in the CL domain are classified as shipper, carrier and receiver (Taniguchi & Thompson, 2002). Nevertheless, this role-based classification can’t fit well here for its ‘multiple inheritance’ (i.e. a class has multiple superClasses) it implies. For example, a retail chain organizing logistic activities on its own can play all the three roles (i.e. have three superclasses) simultaneously (Quak, 2008). According to Horridge, Knublauch, Rector, Stevens, and Wroe (2004), this situation should be avoided in a manually constructed hierarchy (i.e. asserted hierarchy) to mini-
mize the human effort as well as errors in maintaining a multiple inheritance hierarchy. Thus, here we’d better start with a more ‘deterministic’ classification only implying a single inheritance. The hierarchy with multiple inheritances then can be generated afterwards by the logic-reasoner\(^7\).

All the ‘deterministic’ stakeholders listed will be automatically assigned into the ‘Shipper’, ‘Carrier’ and ‘Receiver’ in light of their unique attributes. The rationale of doing this seemly repetitive work is that the objectives/interests of private actors actually come from the role they are playing. For example, an independent retailer acting merely as a ‘Receiver’ will not care about traffic congestion because he/she does not consider the transport as its own responsibility (Quak, 2008). While, a retail chain, on the other hand, knows and cares more about the transport issue because it also acts as a shipper who is well informed about the relation between congestion and logistics costs. Moreover, performable countermeasures and activities also hinge on the roles. Therefore it’s necessary to have a role-based view besides the deterministic categorization.

Thus, role-based stakeholder classes ‘Carrier’, ‘Receiver’ and ‘Shipper’ are added directly under ‘Private_actor’, where these classes are ‘defined class’ equipped with ‘necessary and sufficient conditions’. Now, other classes that meet the ‘necessary and sufficient conditions’ of the “defined class” will be automatically considered to be ‘kind of’ (i.e. subclass of) the defined class in GenCLon. This reasoning is exactly the key to fulfill the task stipulated above, i.e. assigning the subclasses of ‘Determistic_private_actor’ (e.g. 3PL carrier, retailer, inhabitant, etc.) into role-based classes in light of their intrinsic attributes. The hierarchical structure of Stakeholder class is presented in Fig. 3.

4.2.2. Objectives

All stakeholders in the CL domain are driven by their own objectives. Thus, when stakeholders are identified, the next step is to categorize their objectives or interests which can ultimately influence the urban freight movement pattern. All stakeholders involved in the CL system have their particular or shared objectives/interests (Browne, Piotrowska, Woodburn, & Allen, 2007). Some of these objectives are consistent with others, while others are contradictory with others. Generally, objectives within a CL domain can be classified into economic, environmental and social categories.

It is evident from Fig. 4. that these three categories of objectives are interlaced. For example, ‘competitive retail industry’ is facilitating by congestion reduction, nuisance reduction and valuable area protection. An additional point deserving a mention is that ‘efficiency’, a prevalent objective that can be found in many literatures, has not been included here for its ambiguity. It has to be questioned that what ‘efficiency’ exactly is? A rational answer in CL context could be ‘delivering the same amount of goods with fewer vehicles, less vehicle kilometer and time’. For public sectors and citizens, this main’s better accessibility of cities, less fuel consumption as well as reduction in negative influence on the environment and city habitants. For commercial private actors, on the other hand, it will be more related to monetary benefit such as transport or receiving cost reduction. It’s thus convincing to assert that ‘efficiency’ has already been represented by other concrete objectives listed in this section.

In a tree-like structure of the class ‘Objective’ (see Fig. 4.), each link between a subclasses and its superclass has been labeled ‘is-a’ with an arrow going from subclass to superclass. This phrase explicitly indicates that in an ontology, a subclass is ‘kind of’ its superclass and will inherit all the attributes of its parent. Confusion between ‘kind of’ and ‘part of’ must be avoided while constricting hierarchy.

4.2.3. KPIs

KPIs are utilized to measure the extent to which objectives of stakeholders are fulfilled. From a common perspective, ‘KPI’ is a ‘sub-system’ of ‘Objective’ and thus should be put underneath it. However, due to the abovementioned semantic implication (i.e. subclass of superclass), we can’t define ‘KPI’ as a subclass of ‘Objective’. A solution then is to build ‘KPI’ as an independent class parallel to ‘Objective’. The missing ‘part of’ relation can be reclaimed via appropriate ‘object property’ which will be discusses in Section 5. The hierarchical structure of class “KPIs” is presented in Fig. 5. As we can see that no further classification of social indicator is carried out. The reason being indicator for health condition, safety condition and livability of city inhabitant are combined effect of one of the subclass of environment and economic indicator.
Thus, in application using GenCLOn, a correlation among these indicators can be established to estimate the social indicator.

Mirroring the class ‘Objective’, the structure of ‘KPI’ is simpler and only contains two sublevels. The reason is just to respect the nature of terms. For example, the lower level components of ‘Emission_indicator’ are the concentration or mass of all kinds of emission, which can hardly be further classified and thus should be created as ‘instance’.

4.2.4. Resources

Compared with the classes created before, ‘Resource’ involves many hierarchical levels (see Fig. 6). Generally, resource could be classified as ‘Monetary_resource’ and ‘Non_monetary_resource’ where the former comprises of ‘Private_fund’ and ‘Public_fund’, and all the remaining resources fall into ‘Non_monetary_resource’. ‘Non_monetary_resource’ consists of seven subclasses. Starting with “Base”, subclasses of “Base” mainly acts as the destination.
and origin of 'Transiting' in this ontology. Someone may argue that why not directly represent destination and origin by 'Private_actor' (i.e. create a slot 'address' for 'Privator_actor'). This is tempting to do because it actually can simplify the ontology and simplifying is always something desirable. However certain private actor such as a retail chain playing multiple roles in the final distribution of a logistics chain can simultaneously possess at least one depot and one retail premises, and an implied problem will then be: a 'Transiting' can has the same stakeholder as both origin and destination (i.e. depart from address A to address A), which does not make sense. Hence we have to separate the 'location' concept from ‘Private_actor’ and the introduction of 'Base' can exactly fulfill this task. Within the scope of this ontology, the concept of base is that shippers, receivers and UCC (Urban consolidation centre) operators must have certain physical locations where the goods can be stored, handled or delivered.

Notice that, under the class 'Depot', subclasses as 'Warehouse' and 'DC' are attached. Also 'Retail_premise' and 'Other_premise' haven't been further classified because their subclasses don't imply any functional differences (i.e. share the same attributes/properties). Moving to the class of 'Equipment', six subclasses are defined under it, among which 'Freight_vehicle' takes a big share due to the special focus on it. Assuming that most urban freight deliveries are carried out using road freight vehicle, 'Road_freight_vehicle' is further categorized into seven subclasses. It's obvious that these classes are not mutually exclusive and the reason of grouping in this way is to provide various insights. For example, sometimes the users of ontology care about multiple criteria of one object, then monotonous classification will weaken the instructive power of the ontology. All of these have been set as 'defined' classes' and the corresponding 'necessary and sufficient' conditions are based on their key properties. It deserves a mention that a case of 'multiple inheritances' is present here as well, i.e. besides a subclass of 'Base'; 'UCC' is also a subclass of 'Infrastructure'. This situation exactly represents the fact that a 'UCC' can be considered both a base and a kind of infrastructure within the scope of this ontology.

4.2.5. Measures

Different from all the classes given before, there is so far no clear classification of CL measures and thus the hierarchical structure of class "Measure" has to be built from scratch. Keeping the consistency, we still take a stakeholder-based view to group those measures according to their potential implementers. Three top classes are then mapped as 'Governance_measure' which can be implemented by public authorities, 'Private_measure' that can be implemented by private actors and other measures that either need a joint effort of both sides (i.e. PPP) or can be performed by either side (i.e. 'Using_ITS' and 'Using_alternative_mode'). Fig. 7. shows the hierarchical structure of class "Measure" with its first-level subclasses.

**Governance_measure**: This class is defined as 'measures that can be implemented by public authority'. Nine subclasses are formed beneath 'Governance_measure'. Details about this measures can be found in Visser, Binsbergen, and Nemeto (1999), Taniguchi et al. (2004), Browne et al. (2007), Crainic (2008).

**Using_alternative_mode**: This measure focuses on using non-tradition way (e.g. canal) for CL activities. This needs a strong involvement of all the actors that include both public and private actors from activeness and regulatory view point respectively. So, it's actually both a 'Private_measure' and 'Public_measure'. The reason for listing it parallel to its potential subclasses is again the implied multiple inheritance.

**Public_private_cooperation**: As suggested by its name, this measure will never work in the absence of either public or private sides. Thus, following the same principle as above, we isolate it from both 'Private_measure' and 'Governance_measure'.

**Using_ITS**: Actually, this part can be completely assigned to 'Private_measure' and 'Public_measure' without any multiple inheritances. The reason of creating separate class is to highlight the importance of ITS concept in the CL domain. Moreover, due to its inheritance, with the assist of the logic reasoner, they can still be easily assigned into 'Public_measure' or 'Private_measure'.

**Private_measure**: This class is defined as 'measures that can be implemented by private actors'. Different from 'Governance_measure' that enjoys many existing classifications given in literatures, the structuring of 'Private_measure' is exposed to a lack of reference. A casual classification, on the other hand, will negatively influence the accuracy of the ontology. Accordingly, the optimal solution will be directly putting all private measures collected from diverse literature – see review Anand, Duin, and Tavasszy (2011) – under the class 'Private_measure' without any manually asserted hierarchies (except the ones that are very explicit and certain). When 'properties' or 'attributes' are attached to these measures, we can again use the logic reasoner to achieve various categorization. For example, a new class 'Congestion_reducing_measure' can be created under 'Private_measure', and we can give it a 'necessary and sufficient condition' as 'Private_measures' that can help to achieve 'Congestion_reduction' objective. In this way, all measures that meet this criterion will be automatically asserted as a kind of 'Congestion_reducing_measure'.

Fig. 6. Hierarchical structure of class 'Resources'.

Fig. 7.
4.2.6. Activity

Private actors involved in the supply chain perform a series of sequential logistics activities to achieve successful goods movements. The end of an activity could trigger a different or repeated activity depending on the particular situation. Thus, it is possible to roughly reproduce the typical process flow of urban freight transport with these activities and this concept is deemed to be able to facilitate the relevant modeling work. All activities should be performed by at least one stakeholder with certain resources (e.g. equipment, personnel), and sometimes the corresponding infrastructure must be in place (e.g. road for 'transiting'). Main activities involved in a delivery trip within the scope of this ontology are depicted in a tree-like structure below in Fig. 8 (left).

Among these activities, 'Transiting' is highly specified with all the OD pairs within the scope. The reason of doing so lies in the various extents to which the importance of these 'Transiting's can be. For instance, the 'Transiting's to urban premises and homes deserve more focus than the ones between depots for the different geographical areas involved. Correspondingly, there is an extra 'defined' subclass attached under 'Transiting', namely 'Transiting_to_receiver' (see right side of Fig. 8). The necessary and sufficient condition of it is, if expressed in natural language, 'transiting that has urban premise or home as its destination'. This class could be deemed as the 'VIP' of 'Transiting's and is highly relevant to the KPIs of congestion, nuisance and logistics costs. It also has an exclusive attribute as 'extra VKT (Vehicle kilometer travelled)' for finding parking area, which is of poor importance for other 'Transiting's. Besides, 'Loading', 'Unloading' and 'Storing' also get further classified in light of the various physical locations where they can be performed. The reason of doing this is twofold. At first, just like 'Transiting', the locations can determine the potential impact of these activities on the KPIs of CL. Secondly, activities with clarified location can facilitate defining the 'trigger' relations between each other. Thanks to this, a more unambiguous process flow can be represented by the ontology. 'Breaking' and 'Consolidating' are kept alone since they can only be performed in UCCs in the given scope of GenCLOn.

4.2.7. R&D

The hierarchy of 'R&D' is derived in light of the classification proposed by Quak (2008). Although not very old, the CL research domain has been very active since last decade and half and direct effect of this can be found in Fig. 9, which depicts detail classification of R&D approaches available in CL domain. Most of these 'R&D's can directly help to improve the measures or resources covered in this ontology. For instance, the 'Cooperation_R&D' is dedicated to improve the feasibility as well as benefit of cooperation between 3PL carriers, while 'Load_unit_R&D' is self-evidently aimed on developing and improving standard load units suitable for urban freight transport. Differently, 'Research_oriented_contribution' is exclusively aimed to facilitate the R&D work itself and has no immediate relation with certain measures or resources. The whole class of 'R&D' should act as a 'library index' where domain-specific literatures and other efforts can be recorded and categorized.

4.2.8. Value partition

In GenCLOn, 'Fuel' is created as an 'object' rather than an attribute of the class 'Road_freight_vehicle' because we want to specify it in detail via attaching attributes to it as well as further classifying it. It's the same case for 'Base' which is created as an independent object rather than an attribute of 'Private_actor'. In an ontology, properties/attributes can only be attached to 'object' (i.e. class and instance/individual) and this rule implies that all the concepts that require further specification or will be used to describe other objects (e.g. 'KPI' is used to describe 'Objective') must be created as objects. However, some objects to be created don't enjoy positions under the top seven classes defined so far. For example, we want to create a class 'Inventory_policy' to address the inventory policy adopted by a shipper. This new class, however, can be put nowhere under the top seven classes (i.e. 'Inventory_policy' is not a 'kind of' 'Stakeholder' or 'Objective' or 'KPI' or 'Resource' or 'Base' or 'Activity' or 'Measure' or 'R&D'). It's different from the case of 'Fuel' which can be located under 'Resource' because it's actually a 'kind of' resource in the given context. On the other hand, the ontology can easily lose its hierarchy...
and then deviate from the real world (Noy & McGuinness, 2001) if all those 'homeless' classes are created in the same way as 'KPI' and 'Base', regardless of their relative importance in the domain. To cope with this problem, the class 'Value_partition' is created specially to take in those 'homeless' classes without implying too much structural change. The reason of naming it this way is that all the members of this class actually act as certain 'value' used to specify other object via the corresponding 'object property' which will be addressed in detail in the next chapter. The layout of the class 'Value_partition' is given in Fig. 10.

5. GenCLOn axioms

The hierarchical city logistics classes and their instances alone cannot provide sufficient information needed to understand the city logistics domain. Thus the properties describing the attributes of individuals have to be defined. Classes, typically, have two kinds of properties, namely 'object property' and 'data property'. The former could be considered bridges connecting classes/individuals and the latter is actually a defined slot where data value can be filled in. Together with the customized facet and quantifier/cardinality of properties, we can define axioms that stipulate the relationships among classes/individuals are carried out. In simple terms, an axiom is a first-logic expression comprising of subjects, objects and predicates (i.e. properties) in the form of subject–predicate–object. Thus, basically these axioms are used to associate class and property identifiers with either partial or complete specifications of their characteristics, and to give other logical information about classes and properties. Additionally, axioms place constraints on sets of individuals (i.e. classes) and the types of relationships permitted between them. In doing so, it provides semantics by allowing systems to infer additional information based on the data explicitly provided. Staab and Maedche (2000) lists major axiom categories as follow:

- Axioms for a relational algebra: reflexivity, irreflexivity, symmetry, asymmetry, antisymmetry, transitivity, inverse.
- Composition of relations.
- (Exhaustive) Partitions.
- Axioms for subrelation relationships.
- Axioms for part–whole reasoning.
- Nonmonotonicity.
- Axioms for temporal and modal contexts.

5.1. GenCLOn axioms: naming conventions

We follow the same naming convention as we did for defining classes in previous section. Moreover, in case of naming the object

![Fig. 9. The structure of class 'R&D'.](image-url)
property we use the name of the class associated with small modification. For example, we can simply create an object property named as 'has_objective' or 'is_objective_of' to show the connection between 'Stakeholder' and 'Objective'. Then we have two (software-specific) first-order-logic expressions as below:

\[
\text{Stakeholder has_objective Objective} \\
\text{Objective is_objective_of Stakeholder}
\]

With these definitions, object ‘Stakeholder’ and ‘Objective’ now are connected. Notice that there could be various alternative ways to name the object properties above, such as ‘own_objective’ instead of ‘has_objective’, ‘belong_to’ rather than ‘is_objective_of’. Despite that how the property is named will not imply any semantic changes; we are still inclined to adopt a unified naming manner as depicted below to avoid confusion:

- For object property specifying the proactive relations from ‘Stakeholder’ to other classes (e.g.) ‘Objective’, ‘Resource’, naming it in the manner of ‘has_object’ as much as possible, and name the corresponding inverse property as ‘is_object_of’ as much as possible.
- For data property, directly naming it (e.g. address).

5.2. Top-level hierarchy of GenCLOn

In the previous chapter, seven top classes (excluding value partition) have been identified. There are general relations (object property) among these classes and each of them has its own properties that are more specific. Fig. 11 illustrates the main possible relations among the eight classes. Notice that the naming of these properties has been simplified for readability. More specific properties are involved in the lower-level structure that cannot be directly shown in this graph. Fig. 12 and Table 2 show depth and width of city logistics ontology, respectively, in terms of count and visualization which allows estimation of GenCLOn's extent and extension. In the following sections, properties of ‘Stakeholder’ together with part of its subclasses will be given stepwise so as to throw light on some basic axiomatic syntax. To make the content more understandable, some over-complicated syntactic issues will be weakened or skipped. After the necessary introduction, no more explanation is given in the main text to avoid the redundant repetitiveness.

5.3. Properties of GenCLOn classes

5.3.1. Property of top level – ‘Stakeholder’

The properties of the top-level class are the most generic ones and will be inherited by its descendant classes with no exception. In other words, the subclasses of ‘Stakeholder’, namely Private_actor and Public_authority will have all the properties of ‘Stakeholder’. It’s a primary rule of the ontology and is also a reflection of real world.

We start with the property has_objective again and give more detailed explanation. At first, each member of ‘Stakeholder’ must have at least one ‘Objective’. Thus it is rational to assert that

\[
\text{Stakeholder has Objective some Objective}
\]

where, the object property ‘has_objective’ indicates the proactive relationship from the individuals in class ‘Stakeholder’ to the ones in class ‘Objective’. It is clear that in this ontology, only ‘Stakeholder’ can have ‘Objective’ and it is thus with no problem to set the ‘domain’ of the property ‘has_objective’ as ‘Stakeholder’ and the ‘range’ of it as ‘Objective’ (see Fig. 13). Notice that the word ‘some’ here is the so-called ‘existential quantifier’ which indicates that ‘any’ instance of the class ‘Stakeholder’ participates at least one relationship along the specified object property ‘has_objective’ to instances that are members of the class ‘Objective’. The common mathematic expression of the existential quantifier is the symbol ‘\(\exists\)’. Its natural expression exactly equals to ‘each stakeholder has at least one objective’.8

---

8 Since ‘Objective’ is something very subjective, we here only define it by default.
Driven by the same principle, we will similarly derive another stakeholder-related object property as 'has_measure'. An axiom built with the property and the appropriate subject, object and quantifier can then be carried out like below:

\[
\text{Stakeholder has_measure some Measure}
\]

Though here the object property 'has_measure' is created and attached to 'Stakeholder', it is just used to indicate the generic relation between 'Stakeholder' and 'Measure'. The detailed relations between specific stakeholders and their specific measures will be built via its inverse property 'is_measure_of'. The reason is that if the object property 'has_measure' is adopted, then for each subclass of 'Stakeholder' we have to go through all subclasses of 'Measure' in order to choose the ones the specific stakeholder can perform. It's very likely to miss certain measures due to the large number of options (i.e. subclasses of 'Measure'). However, if we do it the other way around, the possibility of missing out will decrease a lot since there are fewer subclasses of 'Stakeholder'.

'Stakeholder' must have a 'name' with which other stakeholders can interact with him. In reality, the 'name' of one 'Stakeholder' could be various, thus the quantifier of it could be 'some'. Then we have to think about the data type of the slot 'name'. In Protégé, there is a complex set of pre-defined data types and some of them (e.g. time, Name) can't be read by the logic reasoners. To ensure a high compatibility and meanwhile avoid work beyond the scope, we hereby only use common data types like 'integer', 'float' and 'string'. In case all of these three don’t fit, the data type 'PlainLiteral' will be adopted and it is compatible with all text typed manually. Obviously, here 'string' should be the one best fit a 'name'. Then an axiom is carried out as below:

\[
\text{name some string}
\]

Besides 'name', the class 'Stakeholder' does not have any more fundamental data properties in this ontology.

5.3.2. Property of second level – 'Private_actor'

In an ontology, on one hand, every subclass will automatically inherit all properties from their superclasses, on the other hand, they will have their own specific properties which their parents don't have. The only constraint is that these properties must be consistent with the properties of their superclasses. For example, we can only assert 'a wine must contain alcohol' and can't assert the 'color' of all wines must be red or white because 'color' is a specific property of the children of 'wine'. Again, since it has been asserted that 'all wines must contain alcohol', then it is forbidden to have a wine 'without alcohol' as the subclass of 'wine'. Due to the highly heterogeneous nature of private actors (e.g. different objective, different resource), there is hardly any shared attributes which can be generally asserted. For the same reason, we avoided to assert constructive properties that could be inherited by all members of 'Private_actor'.

5.3.3. Property of second level – 'Public_authority'

Compared with the heterogeneous nature of individuals of 'Private_actor', properties of members of 'Public_authority' are highly homogeneous and can thus be generally asserted.

No matter local, regional or national authorities, as per their role for people well-being, all want to improve the life quality of urban inhabitants and it’s thus rational to assert:

\[
\text{Public_authority has_objective some Emission_reduction}
\]
\[
\text{Public_authority has_objective some Other_nuisance_reduction}
\]
\[
\text{Public_authority has_objective some Congestion_reduction}
\]
\[
\text{Public_authority has_objective some Safety_objective}
\]
\[
\text{Public_authority has_objective some Health_objective}
\]
\[
\text{Public_authority has_objective some Liveability_objective}
\]

'Fossil_fuel_consumption_reduction' is not included above because it’s a higher-level objective that only belongs to national and even international (e.g. EU) authorities. Roads with different hierarchies are managed by the public administrators on different

---

* Protégé is the ontology building software used for GenCLOn. For more detail [http://protege.stanford.edu/](http://protege.stanford.edu/).
levels. For example, the backbones of a national road network should be managed by the national administrators while local streets are under control of municipalities. Thus it is rational to assert

\[
\text{Public\_authority} \, \text{has\_resource} \, \text{some\ Road}
\]

Surface infrastructures such as parking lots or loading zones in urban areas are managed only by local authorities, and thus will not be set as a property of Public\_authority (the superclass of 'Local\_authority'). Each level of public authorities should have some governable fiscal revenue that could be used to invest in CL projects or to subsidize private actors. Its' thus rational to assert

\[
\text{Public\_authority} \, \text{has\_resource} \, \text{some\ Public\_fund}
\]

5.3.4. Property of third level – ‘Carrier’, ‘Shipper’, ‘Receiver’

Carrier: ‘Carrier’ is the subclass of ‘Private\_actor’ and is also one of the three role-based stakeholders. All carriers have certain homogeneous attributes which can thus be generally asserted. As mentioned in Section 4.2.1, ‘Carrier’ has been set as a ‘defined class’ with a sufficient and necessary condition as below

\[
\begin{align*}
&\text{has\_resource\ some\ Road\_freight\_vehicle} \quad \text{and} \quad \text{has\_resource\ some\ Driver}
\end{align*}
\]

Notice here the term ‘and’ is used to build a ‘composite’ axiom. It can be simply read as its natural meaning, i.e. denoting a coordinative relation. With this axiom, all the private actors that have road freight vehicle and driver as their resources will be automatically assigned into ‘Carriers’ by the logic reasoner. Within the CL context, object properties for carrier are mentioned in below

\[
\begin{align*}
&\text{Carrier\ has\_activity\ some\ Transiting} \\
&\text{Carrier\ has\_activity\ some\ Loading} \\
&\text{Carrier\ has\_activity\ some\ Unloading} \\
&\text{Carrier\ has\_activity\ some\ Off\_board\_transiting} \\
&\text{Carrier\ need\_resource\ some\ Road} \\
&\text{Carrier\ need\_resource\ some\ Surface\_infrastructure} \\
&\text{Carrier\ has\_objective\ some\ Congestion\_reduction} \\
&\text{Carrier\ has\_objective\ some\ Transport\_cost\_reduction} \\
&\text{Carrier\ has\_objective\ some\ Logistics\_quality}
\end{align*}
\]

Given the context of CL, carriers are featured by their unique resources as drivers and road freight vehicles. Hence it’s necessary to know their (instance) available transport capacity which is defined as a combination of available drivers and available vehicles capacity in terms of volume or mass for specific types of goods\(^{10}\) in this ontology. For instance, given a specific type of goods, a carrier has two trucks available with a capacity of five tons each, and only one driver available. Then we can only say that the carrier just has a capacity of five tons available for the time being. The static overall capacity of a carrier may be high, while only the available part makes sense. Thus it’s necessary to assert

\[
\text{available\_transport\_capacity\ exactly\ 1\ float\ [ > = 0.0f ]}
\]

Notice here the term ‘exactly’ is used. It is the so-called cardinality restriction that specifies the exact number of relationships that an individual must participate in Horridge et al. (2004). Accordingly the axiom above could be read as ‘there must be exactly 1 value for the slot ‘available\_transport\_capacity’. The following \([ > = 0.0f ]\) is a software-specific syntax that constrains the range of the filler. Since it is possible for a carrier to temporarily have no vehicles or drivers at hand, ‘0’ should be included in the range.

Carriers (and also shippers and receivers involved in the logistics chain) must provide contact information such as phone number or e-mail address. Thus it is rational to assert:

\[
\begin{align*}
&\text{Contact\_info\ some\ PlainLiteral} \\
&\text{Shipper\ has\_objective\ some\ Logistics\_quality} \\
&\text{Shipper\ has\_objective\ some\ Transport\_cost\_reduction} \\
&\text{Shipper\ has\_activity\ some\ Validating\_goods\_order} \\
&\text{Shipper\ has\_inventory\_policy\ exactly\ 1\ Inventory\_policy}
\end{align*}
\]

Notice that here ‘Inventory\_policy’ is created as an ‘object’ rather than a data slot. It is put under the class ‘Value\_partition’ with four subclasses under it, namely ‘EOQ’, ‘Fixed\_period\_approach’ and ‘JIT’. The reason of doing so is just the same as mentioned before: ‘Inventory\_policy’ is considered an important concept in CL domain and deserves further specification with more attributes. For example, ‘EOQ’ now has data properties as ‘inventory\_level’, ‘order\_quantity’ and ‘ reorder\_point’ which are the three key parameters for this policy. The users of the ontology then can be well informed about this knowledge.

Receiver: ‘Receiver’ is the last ‘role based’ stakeholder, mainly includes ‘Premise\_retailer’ and ‘E\_shopper’. The essential attribute characterizing a member of ‘Private\_actor’ as a ‘Receiver’ is different from the ones for ‘Shipper’ and ‘Carrier’. A receiver does not have any unique resource such as the road freight vehicle and drivers monopolarized by carriers or the goods monopolised by shippers. Thus we have to consider it differently. Apparently, among the three roles involved in the final distribution of a supply chain, only ‘Receiver’ demands the goods to be delivered. It is thus rational to define all individuals that meet the condition below as the members of ‘Receiver’.

\[
\begin{align*}
&\text{need\_resource\ some\ Goods} \\
&\text{Receiver\ has\_objective\ some\ Logistics\_quality} \\
&\text{Receiver\ has\_objective\ some\ Transport\_cost\_reduction} \\
&\text{Receiver\ has\_activity\ some\ Receiving}
\end{align*}
\]

The most important object properties of receiver are listed below

\[
\begin{align*}
&\text{Receiver\ has\_objective\ some\ Logistics\_quality} \\
&\text{Receiver\ has\_objective\ some\ Receiving}
\end{align*}
\]

Premise retailers have clear inventory policies, while it’s tricky to define the way individual consumer order goods via e-commerce. They order goods whenever they want and this behavior pattern actually resembles the ‘JIT’ concept (i.e. small or zero

---

\(^{10}\) For different types of goods, the transport capacity may vary. For example, a carrier may have trucks available for goods with no temperature demand, while have no trucks available for frozen or chilled goods.
inventory, order when need). Thus, if we take a broader definition of ‘inventory policy’, all receivers then could be deemed to be driven by certain inventory policies. This concept will lead to the assertion below:

Receiver has_inventory_policy some Inventory_policy

5.3.5. Property of third level – ‘Local_authority’, ‘Regional_authority’, ‘National_authority’

Local_authority: ‘Local_authority’ is the subclass of ‘Public_authority’ and only has three specific object-property-based axioms as below

Local_authority has_objective some Competitive_retail_industry
Local_authority has_objective some Creation_of_job
Local_authority has_objective some Infrastructure_protection

Each local authority wants that the retail industry in its own area can attract more shoppers, say, mainly from its adjacent areas. It is thus a very local-specific (i.e. selfish) objective that cannot be attached to the higher-level authorities. This rule also applies to the second objective ‘Creation_of_job’ that need to be based on the fulfilment of the first one. For the third axiom, due to fact that the ‘infrastructure’ here only refers to the road in a specific urban area, it thus has to be deemed as an exclusive attribute of local authorities despite that authorities on other levels also want a better use of infrastructure.

Regional_authority: from the stand point of view of regional authority, it is essential to find collaboration for city logistics activities among cities of region. Thus, important property for this class would be

Regional_authority has_objective some Logistics_quality

National_authority: different from lower-level authorities, national administrators have to consider sustainability from a global perspective. Accordingly the preservation of nonrenewable resources comes into agenda. It is thus rational to assert

National_authority has_objective some Fossil_fuel_consumption_reduction

So far, the relations occurring most frequently in this ontology have been illustrated with the axioms of ‘Stakeholder’ along with its top two layers of subclasses. The construction of other axioms follows more or less the same rules and also due to space constraints other axioms are not specified here. The research question as ‘what are the interfaces among objects/entities’ has actually been answered, which implies that the basic elements for an ontology to function properly are satisfactorily covered.

6. Validation of city logistics ontology

There exist many different methods and techniques to validate ontology. Brank, Grobelnik, and Mladenic (2005) carried out survey for different methods available for ontology evaluation and validation where they classify different approaches in following categories:

- Evaluation on the lexical/vocabulary and concept/data level.
- Evaluation of taxonomic and other semantic relations.
- Context-level evaluation.
- Application-based evaluation.
- Data-driven evaluation.
- Multiple-criteria approaches.

To validate GenCLOn we consider data driven evaluation approach. In this approach an ontology is evaluated by comparing it to existing data (usually a collection of textual documents) about the problem domain to which the ontology refers. In this approach, a set of relevant domain-specific terms are extracted from the corpus of documents, using latent semantic analysis. The amount of overlap between the domain-specific terms and the terms appearing in the ontology (e.g. as names of concepts) can then be used to measure the fit between the ontology and the corpus. Here, we consider one agent-based model and two real-life case studies on city logistics for validation purpose.

We, at first, use an Agent based model (Kolck, 2010) to check the compatibility of city logistics ontology for modeling purpose. This comparison will check as to what extent the ontological structure mirrors the model structure. After that two CL projects from real life will be borrowed to further check if the ontology can comprehensively represent abstract the real world. Two criteria proposed by Yin (2003) for case selection are borrowed here. Each case needs to offer an extreme and/or unique situation and must be researched by the same study protocol, namely ontology here. Besides, two customized criteria are also followed. The model case should cover major stakeholders and activities of urban freight transport. For the real-life cases, it is better to have cases carried out in different countries so as to ensure a high commonality.

The model developed by Kolck (2010) is based on cost-valued choices by individual agent and presents results in financial and environmental impact. Six types of agents (i.e. 3PL carrier, UCC operator, truck, shopkeeper, municipality and road) are modeled in this model and all of them are equipped with specific properties. For the validation, agents and their properties in model are compared with their counter parts in GenCLOn. Due to space constraint, only a part of the summary (the agent ‘road’) is exhibited here (see the left side of Table 3) to show the overlap of GenCLOn with exiting agent based model.

This validation shows that all of these six agents can find their counterparts in the CL ontology. However, this is still far from convincing because the six agents are commonly used in most urban freight models. More conclusive evidence can be obtained after checking their properties. Among the 38 properties found in Kolck’s model 35 properties have been explicitly manifested or incorporated in the ontology. A big share of them (24 out of 35) can be directly understood from the ontological classes corresponding to their agents in the model (e.g. tariff, toll collected and speed limit of ‘road’, etc). Those which could not be referenced directly are present in the ontology implicitly (e.g. km and toll count in this model are deemed the properties of truck, while in the ontology they are defined as the attributes of ‘Transiting’). The only part that can hardly be represented by the CL ontology is the routing related property, which happens to be the most model-specific as well as dynamic part. Nevertheless it actually does not influence the contribution of the ontology significantly since this part is exactly the core of the model where the developer should have their own profound as well as original interpretation. In this sense, it’s convincing to say that at first the ontology can provide CL modelers with competent information, and secondly the ontological structure mirrors the model structure well. Hence, the primary mission of the ontology as laying a good basis for modeling work has been fulfilled.

Besides this, two real-life case studies are used to validate the CL ontology in order to investigate at what extent it can reflect the real world. The first case is the Binnenstadservice (BSS), a UCC-based CL project starting in April 2008 in the Dutch city
Both validations are steered by the same approach used in the previous modeling case. Project descriptions in the relevant papers are summarized into a series of topics which are in turn associated with the elements in the ontology. Encouragingly, 18 out of 20 topics in the first case, and 17 out of 19 topics in the second case can be either directly or indirectly represented by the ontology. The results of the validation process for these case studies bolster the ability of the CL ontology to comprehensively represent the real world.

7. Application of city logistics ontology

Containing comprehensive knowledge is just the basic characteristic of an ontology, however the key issue lies in how to use this knowledge. An ontology can be used passively as a database where users can acquire knowledge of interest or be informed about the knowledge structure of the domain. It may also be used actively as a component of models and actively interact as a modeling part and thus help to analyze the knowledge it contains.

7.1. Knowledge sharing

The most basic application of an ontology is to share the domain knowledge with its users as information is highly structured in ontology and thus caters to a specific knowledge-acquiring. Users can search for the terms of interest and in turn check their attributes as well as annotations which provide highly refined information. For example, using GenCLOn a user can easily start exploring information about UCC. After selecting the class ‘UCC’ (see Fig. 14), s/he can immediately get the definition of ‘UCC’ via the corresponding comment of the class (see Fig. 15). Besides this, a list of axioms will be also visible under the annotation (see Fig. 16). Important attributes of UCCs are summarized into these axioms and each of them is equipped with dedicated explanation similar to comment for definition. A straightforward classification of the main three types of ‘UCC’ is visible after a click on the small triangle (see the right of Fig. 14).

7.2. Analysing and reasoning

Another important application of ontology is to analyze and reason the knowledge it contains, and this can be achieved mainly via two ways, namely automated categorization and query.

7.2.1. Automated categorization

The logic reasoner (e.g. Protégé – in this case) can restructure the manually asserted ontological hierarchy and in turn infer or discover new/hidden relations based on the manually asserted ones. This is especially useful when an ontology becomes too large as well as complex for human to maintain it. It has been mentioned Section 4.2.1 that those deterministic stakeholders such as retailer, B2B shipper, 3PL carrier and inhabitant would be assigned to the role-based classes ‘Shipper’, ‘Carrier’ and ‘Receiver’ in light of their relevant attributes. This operation is exactly a typical automated categorization suitable to be performed here as an example.

Based on the abovementioned features, the user is very likely to get a quick sketch of what s/he wanted to know very easily. It is also possible to access other related terms via the corresponding axiom (e.g. in the example of ‘UCC’, ‘UCC_operator’ is connected via the object property ‘is_resource_of’), and then a systematic learning course is being practiced subconsciously. What’s more, for ABM developers, the abovementioned function actually provides an easy interface to sort out the relationships between different objects and in turn add their own object instances for specific case studies (i.e. data entry) (Keirstead & van Dam, 2010).

7.3. Table 3

<table>
<thead>
<tr>
<th>Agent</th>
<th>Properties in the model</th>
<th>Relevant content in the CL ontology</th>
</tr>
</thead>
<tbody>
<tr>
<td>Road</td>
<td>has speed limit</td>
<td>Road speed_limit some float[&gt; 0.0f]</td>
</tr>
<tr>
<td></td>
<td>has real speed per time phase</td>
<td>Road instant_road_speed exactly 1 float[&gt; 0.0f]</td>
</tr>
<tr>
<td></td>
<td>can estimate toll collected</td>
<td>Road toll_collected exactly 1 float[&gt; 0.0f]</td>
</tr>
<tr>
<td></td>
<td>can estimate NOx</td>
<td>Transiting performed_at some Road emission_during_transiting exactly 1 float[&gt; 0.0f]</td>
</tr>
<tr>
<td></td>
<td>has different colors to indicate crossing</td>
<td>Road has_infra_component some Junction</td>
</tr>
<tr>
<td></td>
<td>can change the speed to account for traffic conditions</td>
<td>Road instant_road_speed exactly 1 float[&gt; 0.0f]</td>
</tr>
<tr>
<td></td>
<td>can charge toll from trucks</td>
<td>Road tariff some float[&gt; =0.0f]</td>
</tr>
</tbody>
</table>

A ‘Axiom’ here can be simply understood as a first-order sentence composed of property, quantifier and value of the property. In Protégé, axioms are specified under the corresponding classes.

Fig. 14. Expanding the class ‘UCC’.

Fig. 15. Annotation on class ‘UCC’.

Fig. 16. The list of axioms of UCC.
In Fig. 3. Hierarchical structure of class ‘Stakeholder’, it is clear that all the three role-based stakeholders do not have any subclasses at that time. However, after running the reasoner, it can be immediately perceived that there are subclasses newly attached to them (i.e. there are marks in front of them). If we expand these new inferred classes by clicking on the marks, their subclasses will look like Fig. 17. It is self-evident that the classification achieved by the reasoner is more complicated and involves several multiple inheritances. For example, classes such as ‘Retail_chains’ and ‘Independent_retailer’ are simultaneously (completely or partly) assigned to different role-based stakeholders. With this automatically asserted hierarchy, the roles stakeholders can play in the urban freight chain become intuitive.

7.2.2. Query and answering

A well-constructed ontology can help to analyze domain knowledge by answering domain queries. When the objects (classes or instances) in an ontology are specified with attributes, then a series of queries based on those attributes can be posed by users and in turn be answered by the ontology with the help of appropriate software platform (e.g. Protégé – as in this case). One example query is posed here:

“To achieve a specific objective, what measures can be performed by the stakeholders?”

Let us use the objective ‘Road_protection’ as an example. To answer the first sub-question, the ‘DL query’ function provided by Protégé will be used to avoid manual effort (i.e. manually checking each subclass of ‘Measure’). The only premise for using this function is that the input question must be given in the formal language that can be understood by the program. The first sub-question can then be formally expressed as

Measure and may achieve some Road_protection
After a click on the ‘Execute’ button (see Fig. 18), the ‘DL query’ will execute this question automatically on the basis of the predefined properties of each descendant class of ‘Measure’. A list of competent measures will then be displayed as lower part of Fig. 18. The essence of this operation is to pick out all the subclasses of ‘Measure’ that have a ‘may_achieve’ relation with the class ‘Road_protection’.

7.3. Modeling and simulation

The ontology can be used as knowledge sharing platform to understand the concept, its properties and relationships etc. While this is – indeed – useful but might not be sufficient for the modeling purpose. In fact, it would be much more useful if the model can access this information for the building blocks of model and also can query information it requires directly from ontology during runtime. There are multiple Java libraries available for this purpose: Sommer, Elmo, Jenabean, owl2java, RDFReactor etc. Based on this understanding, the usage of ontology for modeling purpose falls in two categories: code generation and annotation-based binding. Java code generator generates java classes using ontology classes, properties and relationship attached with ontology. A code generator can be tremendously useful and time saving, provided the ontology is properly detailed with specific property ranges (Cowan, 2009). On the other hand, in annotation-based binding approach, annotation of java classes are bind with ontology data which allows the instantiated object to acquire relevant data from ontology – class or property. This acquiring relevant data is done using query language (e.g. SPARQL) in java class of the agent. Each approach has its pros and cons, however, the annotation driven approach can be more useful as it allows encoding the agent behavior in Java class methods and then binding to the ontology only to populate the fields of the class instances, which could not be achieved through code generation. Fig. 19 gives overview of these approaches. Examples with detail description about both of this method can be found on web (Giudici, 2009) and literature (Quasthoff & Meinel, 2009).

8. Conclusion

In this paper, an effort has been made to introduce a formal ontology aimed to systematically as well comprehensively specify the domain of CL in terms of the concepts involved along with their relations among each other. Extensive information and knowledge has been collected from relevant literatures as the theoretical foundation of the ontology. After a series of information processing including sorting, refining and summarizing, the domain of City Logistics is then classified into eight general classes, namely ‘Stakeholder’, ‘Objective’, ‘KPI’, ‘Resource’, ‘Measure’, ‘R&D’ and ‘Value_partition’. Together with the large number of subclasses attached afterwards, they represent the city logistics domain with a hierarchical structure that abstracts the real world. Most classes are interconnected with others via defined relations, and are noted with specific attributes that can help to build corresponding instances. After validations based on an existing agent-based model used to simulating performance of urban consolidation center, and two real-life cases based on CL projects initiated in the Netherlands and Spain, the ontology developed so far appears to be competent with providing necessary information and in turn laying a good basis for domain-specific modeling work.

Conceivably, there will never be a perfect ontology since it’s just a representation of knowledge that is something ever-growing and thus infinite. A manually constructed ontology has to be improved and enriched continuously. For example, in the city logistics ontology presented in this paper, some important factors such as public transport, individual shopping trips and service-oriented traffic are weakened considerably due to the delimitation of GenCLOn. Only their ultimate influences on freight traffic are shallowly represented in the ontology via data properties as instant traffic load of roads and demand rate of goods. Also detail logistics concepts (e.g. supplier contract, transportation contract etc.) are not included as the purpose of GenCLOn is to represent daily activities of urban goods movements. Nevertheless, it should be noted that these concepts are very important as they decide pattern of urban goods movement. Further effort is demanded to specify these factors in detail, possibly via introducing new stakeholder (e.g. service provider), new activities (e.g. choosing suppliers, consuming goods) and new measures (e.g. enhancing or creating public transport service), etc. It is also possible to combine these details from other ontological efforts mentioned in literature review.

Besides the inevitable delimiting, the content covered by the ontology can also be further refined as well as enriched. At first, the ontological restrictions on classes and properties have been deliberately simplified to facilitating reasoning and debugging that have been heavily burdened by the large number of objects (i.e. classes and instances) along with their properties in the ontology. For instance, some object properties can be further defined as ‘functional’, ‘symmetric’, ‘transitive’ etc. and their domains and ranges can be strictly constrained. All of these operations will definitely make the ontology more precise, while can also impose high hardware requirements as well as more exposure to inconsistency caused by strict ontological commitments among objects. These inconveniences should be avoided by a newly-developed ontology that has to be debugged from time to time. If competent hardware is in place, further work can be taken to reclaim these restrictions. Besides, some data properties such as ‘engine’ of vehicles can be converted into object properties that is able to be further classified and specified with relevant data properties along with comments. This kind of conversion can even introduce new semantics. An intuitive example is that terms such as ‘reputation’ and ‘address’ can be connected to a new activity (which is created as a class in the ontology) as, for instance, ‘Choosing_supplier’ via a new object property as, for instance again, ‘depend_on’ after the conversion. This formulation will explicitly indicate that reputation and physical location (the distance between the bases of shippers to the receivers’ sites) are two important criteria during supplier-choice. Similarly, almost all the classes can be enriched continuously, and sometimes it is even possible to directly import external ontology. For example, if an ontology dedicated to road is accessible, we can just download it and import it into the city logistics ontology, and substitute it for the current class ‘road’ as long as the new one is more competent. Similarly, part of this ontology can also be directly combined with other relevant ontology (e.g. an ontology for the holistic domain of logistics).

Moreover, due the limitation of the current ontology editors, some dynamic/stochastic relations among objects can hardly be represented. Accordingly, relations defined by default can deviate more or less from the real world. For example, the relation between stakeholders and objectives are quite uncertain while have to be asserted in a deterministic way in the ontology nowadays. Encouragingly, there are now some initiatives on introducing randomness with the assist of approach like ‘fuzzy logic’ (Bobillo, Delgado, Gómez-Romero, & López, 2009; Tho, Hui, Fong, & Cao, 2006). These concepts, nevertheless, are still in a juvenile age and demand much further development. It is believable that an ontology will be able to reflect the world in a more realistic way when these technologies are mature enough.

Finally, this ontology is still in a generic level due to the extensiveness of the domain it tries to represent as well as the relatively low importance of instance-creating at the current phase. As a
result, many classes just end up with sub-classes rather than specific individuals that are addressed with concrete data. For example, in the ontology we only have classes such as ‘Goods’ or ‘Retail_premise’ and don’t have a specific type of goods such as ‘Coca Cola 330 ml’ or a specific shop like ‘Wal-Mart 88th Street’. Fortunately, the ontology itself will act as the template for instance-building since all the attributes that an instance should possess have already been stipulated upon the class it belongs to. The work left is just collecting and assigning data to the corresponding slots. In other words, the ontology stipulates the data required for an instance (e.g. a truck should be specified with load capacity, engine type, etc.). This is exactly one of the most important merits of an ontology, namely facilitating instance-building.

Acknowledgements

We thank four anonymous referees from TRB conference for providing a range of very useful suggestions for improving earlier version of this paper. Support from Netherlands Organization for Scientific Research (NWO) for funding this research (Project: Sustainable Accessibility of the Randstad) is greatly acknowledged.

References
