TOWARDS INTEGRATED ASSESSMENTS FOR URBAN DEVELOPMENT

SUBMITTED: June 2005
REVISED: December 2005
PUBLISHED: May 2006 at http://www.itcon.org/2006/17/
EDITOR: Dana J. Vanier

Hans A.J. Schevers, Dr.
CSIRO Manufacturing and Infrastructure Technology
email: Hans.Schevers@csiro.au and http://www.cmit.csiro.au

Gerardo Trinidad, Dr.
CSIRO Manufacturing and Infrastructure Technology
email: Gerardo.Trinidad@csiro.au and http://www.cmit.csiro.au

Robin M. Drogemuller
CSIRO Manufacturing and Infrastructure Technology
email: Robin.Drogemuller@csiro.au and http://www.cmit.csiro.au

SUMMARY: *The role of sustainability in urban design is becoming increasingly important as Australia's cities continue to grow, putting pressure on existing infrastructure such as water, energy and transport. To optimise an urban design many different aspects such as water, energy, transport, costs need to be taken into account integrally. Integrated software applications assessing urban designs on a large variety of aspects are hardly available. With the upcoming next generation of the Internet often referred to as the Semantic Web, data can become more machine-interpretable by developing ontologies that can support the development of integrated software systems. Software systems can use these ontologies to perform an intelligent task such as assessing an urban design on a particular aspect. When ontologies of different applications are aligned, they can share information resulting in interoperability. Inference such as compliancy checks and classifications can support aligning the ontologies. A proof of concept implementation has been made to demonstrate and validate the usefulness of machine interpretable ontologies for urban designs.*

KEYWORDS: *urban development, semantic web, integrated assessments, software interoperability.*

1. INTRODUCTION

The role of sustainability in urban design is becoming increasingly important as Australia's cities continue to grow, putting pressure on existing infrastructure such as water, energy and transport. Currently regulations and design guides are available to support the design of a sustainable urban design such as AMCORD (1992), energy ratings, building codes, etc. Consulting all these guidelines can be a complex and tedious task. At best, a subset of these criteria can be readily investigated as part of the design phase. The adoption of new, alternative systems in combination with the increased awareness of sustainability adds significant complexity to the design process and as such, determining the appropriate conceptual design for an urban area (either new, green-field development or re-development within an existing area) can be extremely challenging.

Obviously, software automating multi-disciplinary assessments can help to produce more sustainable urban designs (CRC for Construction Innovation, 2005). Stakeholders can inspect the different urban design solutions in virtual reality having relevant information at hand such as costs, water and energy usage, distances, density, etc. Stakeholders can run 'what-if' scenarios on different design solutions to increase their understanding, optimise design solutions, and eventually make informed decisions.

Currently, integrated software systems assessing urban designs on a large amount of aspects are virtually non-existent. However, there is a reasonable amount of software available for assessing particular parts of an urban design. On the building level many software systems exists such as heating, ventilation, air-conditioning, and cooling (HVAC) systems. On the urban scale there are Geographic Information Systems with a variety of functions and data, and software applications dealing with water models for catchments, etc. Most of these models are developed independently and for a specific purpose. This has resulted in many software applications with different scopes using a variety of modelling techniques, different software architectures and computer languages. Integrating these heterogeneous models is therefore a difficult task. In addition, many of these applications will develop overtime (just as regular software) and new models will emerge. A software framework is necessary to make these models interoperable but also allowing the individual applications to develop further. Such a framework should be flexible enough to incorporate new applications without increasing the complexity of the whole system too much.

This paper investigates Semantic Web technology to provide for such a framework in order to facilitate the development of an integrated assessment system for urban development. Section Two gives an overview of Semantic Web. Section Three describes how this technology can be used for an integrated assessment system for urban development. Section Four discusses a proof of concept software implementation demonstrating the feasibility of using Semantic Web technology for this purpose.

2. SEMANTIC WEB

2.1 Overview

Semantic Web (SW) is a term coined by Tim Berners-Lee with the goal to make data on the web more machine-interpretable (Berners-Lee et al, 2001). The main goal is to make the current Internet more intelligent by enabling computers to perform knowledge intensive tasks using the Internet as a resource. The SW provides a standardized framework that allows data to be shared and re-used across applications, enterprise, and community boundaries. A set of technologies forms the basis for the Semantic Web framework. Languages such as XML, RDF (Resource Description Framework) (Brickley and Guha, 2004) and OWL (Web Ontology Language, 2004) enable the marking up of data using declarative statements making the data more machine-interpretable. The difference between these languages is the predefined mark-up statements, which increases the expressive power. Using these languages, documents called ontologies can be created formalising concepts, relationships, etc. in a computer interpretable form by 'marking up' the data. Nowadays more and more knowledge representation languages such as rule languages like JESS (JESS 2005, Friedman-Hill 2003) and CLIPS (2006) support reasoning/inference on these ontologies. The SW ontologies are inherently web-based supporting 'hyper-linking' possibilities. This means that individual concepts in an ontology have a unique web address which can be re-used by any other ontology residing on another web space. Basically concepts can be re-used easily by reference resulting in networks of ontologies which support interoperability.

2.2 Semantic Web Ontologies

Semantic Web ontologies structure the underlying data to make that data machine interpretable and transportable. The Resource Description Framework (RDF) is a standardized language able to identify *things* using web identifiers called Uniform Resource Identifiers (URI) assuring that others can address these things (like a hyperlink). RDF can describe these *things* in terms of simple properties and property values by using XML as syntax using declarative statements. Fig. 1 shows a snapshot of two ontologies that are related to each other using an URI. RDF Schema is a vocabulary for describing properties and classes including generalization-hierarchies and *individuals*, which are instances of classes. OWL comes in different variations and enriches the RDF schema vocabulary with relations between classes (e.g. disjoint relation), cardinality, richer typing of properties, characteristics of properties (e.g. transitive property), etc. One of the variations of OWL is OWL Description Logic (DL) (Baader et al, 2003) supporting maximum expressiveness while retaining computational completeness (all conclusions are guaranteed to be computable) and decidability (all computations will finish in finite time). Using description logic engines, consistency checks and classifications can be performed. For example, when using the 'disjoint' statement between classes it means that an individual cannot belong to both classes at the same time. Obviously these classes cannot have a specialisation relationship between them as the subtype class could never have instances. Consistency checks can find these classes automatically.

Classification can determine the class hierarchy using the logical definition of the classes and membership relations. For example, specialisation hierarchies can be determined automatically, which is very useful for large ontologies or networks of ontologies. In addition, individuals can be classified to which class they belong. This feature also enables the inference of interoperability. For example, consider that a, b and c are information entities that are necessary for different applications to run. When a = b and b = c, then you could infer that a = c. This means that making inferences on the ontologies can result in reusing an information entity.

```
<owl:Class rdf:ID="Lot"/>
<owl:DatatypeProperty rdf:ID="lot_size">
    <rdfs:domain rdf:resource="#Lot"/>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#float"/>
  </owl:DatatypeProperty>
```

Snapshot of ontology on `http://www.cmit.csiro.au/idcs/urbandevelopment.owl`

```
<owl:Class rdf:ID="BuildingSite">
<rdfs:subClassOf  rdf:resource="http://www.cmit.csiro.au/idcs/urbandevelopment.owl#Lot"/>
</owl:Class>
```

Snapshot of another ontology referring to the class Lot defined in the ontology above

FIG. 1: Elements in an ontology document can relate to other elements in other ontology documents, due to the unique addressing of each object using URIs.

A class definition describes their members that are the individuals and therefore contains a list of restrictions. Individuals meeting these restrictions are members of that class. Class definitions such as union, disjoint and cardinality and topological restrictions can be used to define a class. To determine if an individual is a member of a certain class, classification of that individual is necessary which can be done by a description logic engine. This means that different ontologies can share individuals which can be classified to different classes (FIG. 2). Currently restrictions on property values are not (yet) available (e.g. distinguishing high-rise buildings from low-rise buildings by constraining the height is not yet supported.) However classifying individuals based on their topological relationships and characteristics of their properties is possible.

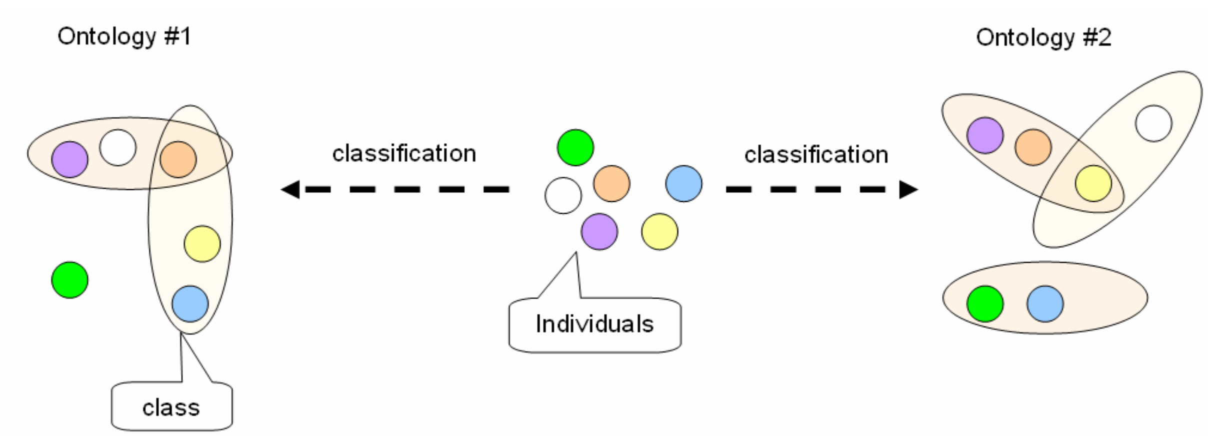

FIG. 2: An individual can belong to several classes.

2.3 Knowledge Representation Languages

According to Motta (1999) a new approach that explicitly distinguishes a universe of discourse and Knowledge Representation Language (KRL) with a Problem Solving Method (PSM) improves the scalability and re-use of Knowledge-Based Systems. A KRL uses a formalism to represent knowledge called a Knowledge Representation (KR) (Sowa, 2000). Ontologies serve as sources of information for a KR. Consequently different knowledge representations can use the same ontology. As SW ontologies are standardized, there are already several knowledge representation languages available capable of inferring on these ontologies. For example, the Protégé project (Protégé, 2005) enables the use of a SW ontology for rule based systems such as Java Expert System Shell (JESS) (Friedman-Hill, 2003), Algernon (2005) and logic programming using Prolog (2005), etc. Already ontologies and rule languages have been proposed for interacting with Geospatial databases (Map Bureau, 2003). Other KRLs such as MathML (MathML, 2002), a low-level specification for describing mathematics as a basis for machine-to-machine communication, are being developed to operate on SW ontologies. In several situations the KRLs can be used together operating on the same model. However the KRL are not interconnected and it is difficult to see which KRL is influencing what in the ontology. To determine a priori which KR influences what in the ontology, standardisation of KRs is currently being investigated such as Semantic Web Rule Language (SWRL 2005) (Golbreich and Imai, 2004). SWRL rules are captured in the ontology and therefore easily transferable. However its current expressiveness is limited. For example, SWRL cannot provide the expressiveness in terms of rules and language for geospatial applications (Chen et al, 2005, Lieberman et al, 2005).

2.4 Service Oriented Architecture

Object Orientation supports the re-use of software and therefore supports the development of complex and large software systems. Service Oriented Architecture (SOA) is a software architecture trying to achieve a loosely coupled network of collaborating and interacting software applications. A service in this context is a unit of work done by a service provider, which can be a software application, agent, a web service, etc. Communication between the services is therefore very important. This enables a SOA service to be functional which means that the architecture does not specify how the service is going to perform the task. This can be perceived as the next level of abstraction to deal with the complexity of large software systems. Individual development of the application performing the service or even a total replacement is possible when the service remains the same. In addition, SOA is extendible and scalable. More services can just join the network, especially when services can join the network dynamically. Eventually these services can be composed dynamically forming ad hoc chains designed for a particular purpose (Fensel, 2002).

SW supports this chaining of web services and uses its machine interpretable ontologies to describe and discover web services on the net. A whole range of communication languages have been developed to support this process such as Universal Description, Discovery and Integration (UDDI, 2005) for registering and finding web services, Web Services Description Language (WSDL, 2005), Web Ontology Language for web Services (OWL-S, 2005), Web Service Modelling Ontology (WSMO, 2005) for describing the web service, Simple Object Access Protocol (SOAP, 2005) to communicate with the web service, etc. (Daconta et al, 2003). Currently many overlapping technologies are available such as WSDL, WSMO, OWL-S, etc. and some of them still are under development or will undergo a revision.

2.5 Discussion

Semantic Web has some interesting characteristics:

- The distributed ontology approach enables the creation of distributed ontology driven software while the ontologies can support interoperability and the usage of different knowledge representations.
- Semantic Web approach uses an open architecture standardizing only the language and protocols rather than content standardization. This means that everyone on the web is able to create content similarly with the current Internet.
- The declarative nature of the ontologies supports inference on ontologies for compliancy checks and automatic object classification and even determining interoperability automatically. This is particularly handy when ontologies are getting complex (when, for example, many different ontologies are being interrelated). In addition, all the intelligence can be formalised in an open and declarative way instead of a (hard coded) black box software application.
- Classification of individuals can support different views and consequently multiple domains with their own class ontology-sharing individuals. In addition, the classification supports reasoning on an evolving individual. Especially during design processes, design information evolves from rough to fine. Individuals can grow adding more properties and relationships. Classification can determine new membership relations to classes that can result in different rules executions, etc.
- Service Oriented Architecture support re-use of services and supports forming ad-hoc chains of services. This architecture enables an evolutionary software approach where new web services can join the network.

3. SEMANTIC WEB FOR INTEGRATED ASSESSMENTS FOR URBAN DEVELOPMENT

3.1 Conceptual Software Architecture

A service oriented architecture using standardized Semantic Web technology can easily be adopted for developing software for integrated assessments for urban development. For example, a generic urban development application can make use of a GIS service and its ontology. Concepts about (footprints of) buildings in the GIS ontology could be extended by the urban development application to hold more information than available in the GIS system. By extending the GIS ontology, use of GIS data such as the geometry of the footprints of buildings is assured. A water usage prediction web service/ application could extend the building concept as well with properties such as *water usage*, *amount of showers*, *building type*, *garden size*, etc. Using this information the water service can estimate a water usage per dwelling. Another application capable of calculating energy consumption can extend the ontology with energy slots and relationships such as *windowtypes*, *walltypes*, etc. Fig. 3 shows a network of software applications forming a collaboration which is possible when the ontologies are interoperable.

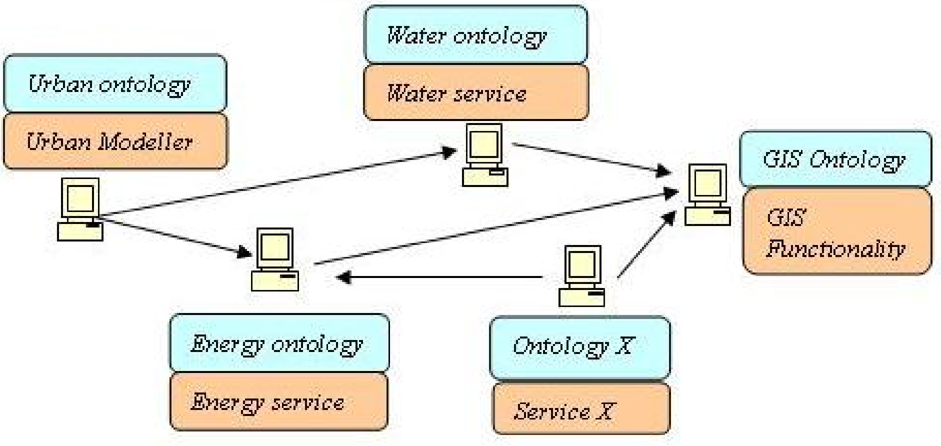

FIG. 3: conceptual architecture for assessing urban development using Semantic Web Technology

3.2 Re-using legacy applications

Legacy applications have been developed which may need to participate in the Service Oriented Architecture. Therefore these applications need to become ontology driven. One approach is to totally re-program the application in Semantic Web compliant technology with the advantage of using standardized knowledge representations. Another approach is externalising the implicit (internal) ontology and making sure the new SW ontology is connected with legacy application (Schevers, 2004). Creating a coupling between the externalised ontology and the original applications enables the applications to participate in the proposed framework (Fig.4).

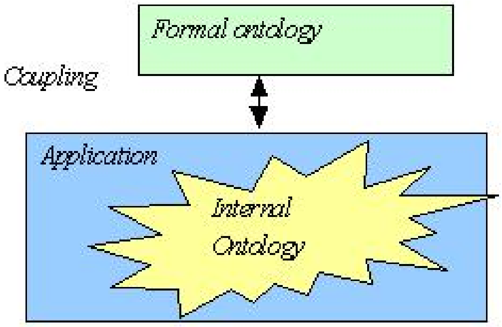

FIG. 4: Externalising ontology from a legacy application so that it can participate in a Semantic Web environment

3.3 Aligning Ontologies for Interoperability

When new and legacy applications have a Semantic Web compliant ontology, these individual ontologies need to be aligned for interoperability. Several ways exists to align these ontologies (Fig.5):

- Use the same Ontology. When two services or applications use the same ontology interoperability is achieved.
- Extending Ontologies. One service (application) based on ontology can be integrated with another service (application) when their ontologies are compliant. This means classes are similar and can be re-used within the applications.

- Super Ontologies. Another approach is to create a super ontology for different services using different ontologies. The Super ontologies act as a basis for integration between both services. Concepts in the super ontology can be shared between the two services. No interoperability is available for concepts defined in the sub ontologies.
- Ontology mapping. Concepts can be related concepts from both ontologies using additional knowledge, i.e. mapping rules. These rules can be used to find concepts in the other ontology that correspond to concepts used in a query and retrieve the instances of these concepts as well. The mapping itself can be seen as a service as well and consequently becomes reusable as well. An example is a unit conversion web service which can translate imperial units to metric units.

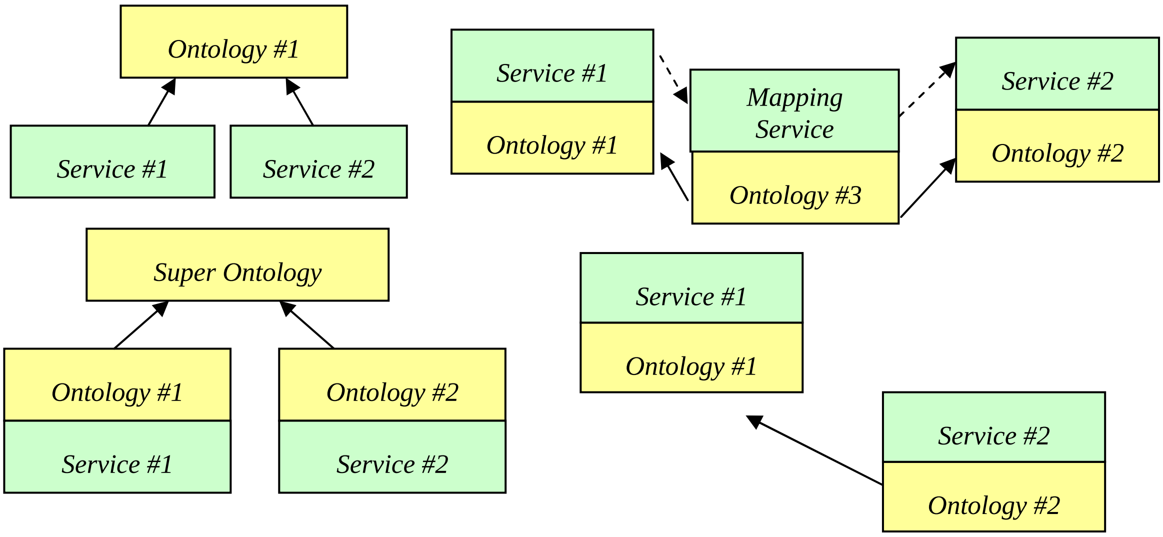

FIG. 5: Several ways for aligning ontologies

3.4 Towards integrated assessments for urban development

Developing an integrated assessment system for urban development from scratch is hardly feasible. An evolving approach is more feasible. Therefore ontology driven software needs to be developed which can be used for urban development. Ontology editors, databases, ontology driven GIS systems, knowledge representation languages and inference engines in combination with urban ontologies and assessment applications are necessary ingredients for developing such an application. However, the Service Oriented Architecture allows starting from a simple ontology driven system, which then can evolve towards a more sophisticated system comprising many other ontology driven applications and web services. The key will be to make sure the ontologies are interrelated and consistent. Therefore interrelations have to be defined by humans who need to oversee at least two ontologies besides having good ontology/modelling skills. Inferring on all the ontologies when several interrelationships have been asserted may result in new inferred relationships. For example, linking the concept *building* in the GIS ontology with the *building* concept in the water *and* energy ontology results in the *building* in the water ontology being similar to the *building* in the energy ontology (FIG. 6). Description logic engines can infer these relationships and therefore support interoperability between applications. In addition, it supports keeping the (related) ontologies consistent.

'building'(GIS ontology) = 'building' (energy ontology)

'building' (GIS ontology) = building (water ontology)

(Inferred)

'building' (energy ontology) = building (water ontology)

FIG. 6: An example how machine interpretable data can support interoperability

4. PROOF OF CONCEPT IMPLEMENTATION

4.1 Implementation

For the implementation of an integrated assessment system for urban development, an open source tool called Protégé is used (Protégé, 2005). Using Protégé classes, slots, restrictions can be defined graphically using OWL as underlying ontology language (Knublauch, 2003) (Fig. 7)

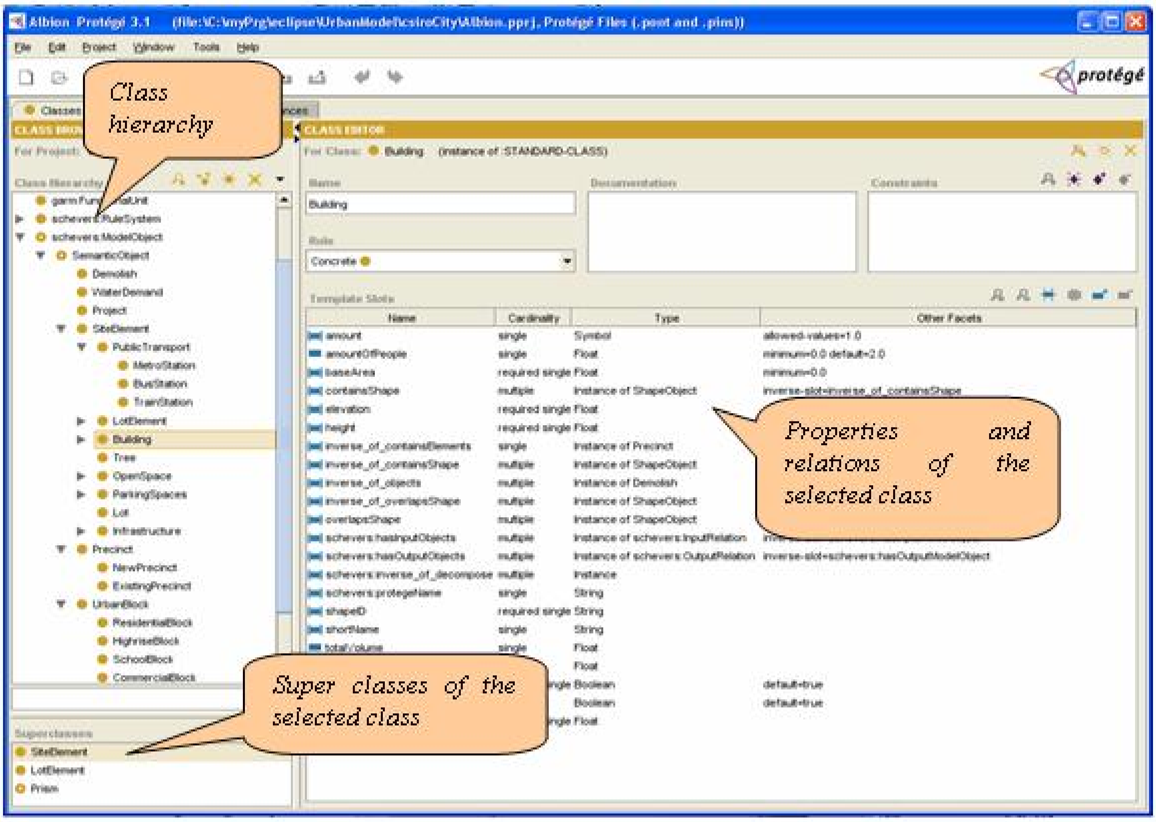

FIG. 7: Screenshot of Protégé

Protégé supports the creation of individuals. For each class, a graphical user interface is created automatically, which enables the setting of values for individuals. As each individual can belong to multiple classes, connections are available to descriptive logic inference engines such as PELLET (2005) and RACER (2005) for classification and consistency checks. To mimic GIS functionality, classes capable of containing shape information have been developed such as a class *polygon* containing *point* with slots x, y, z, representing the coordinates, etc. A 2D user interface has been developed to visualise the geometry captured by these classes. A geometry agent uses the same ontology and makes sure that values such as area and length are set as well. In addition, simple spatial queries such as the distance between two polygons can be made. For adding behaviour to the individuals, a decision table plug-in has been developed that operates directly on the ontology. Decision tables can be defined for each class in the ontology defined in Protégé. A decision table accommodates the development of simple 'if–then' rules in a tabular format (Fig. 8). Each decision table has a conditional part and an action part. In the condition part, the conditions are formulated. When an instance of the class meets these conditions, the actions specified in the same column of the decision table will be carried out. Fig. 8 shows the following example: if the height is 30 or lower than the type, the property is set to 'low-rise'.

Using the geometry ontology and its implementation including rule based systems and the decision table plug-in urban development ontologies can be created re-using the previous ontologies. Rules and decision tables can be developed and re-used individually. Two prototype systems have been developed by creating an ontology and defining the necessary behaviours (decision tables and rules).

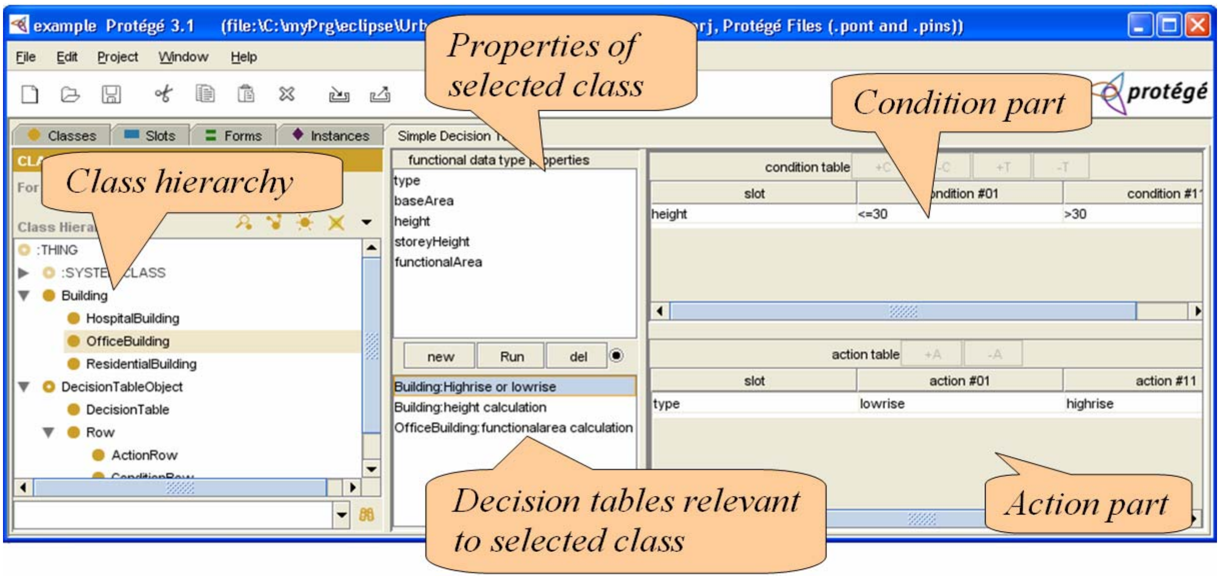

FIG. 8: Screenshot of the decision table plug-in for Protégé

4.2 Urban master plan prototype

4.2.1 The ontology

The ontology of a master plan contains the classes, such as *precinct*, which can be decomposed into *zones* that contain *zone functions*. A *zone* can be decomposed into *zone functions* such as *infrastructure*, *residential*, *park*, etc. Each *function* has a property defining the percentage of the *zone* that is used for that function. The idea is that the user can select a *zone* and insert that *function* it needs to have. Further specifications of each function have been made: for example, the *residential function* has a property house type, which has values (large detached dwelling, semi-detached dwelling, town house etc.).

4.2.2 The rules

The behaviours are of course based on the ontology. The decision table plug-in is used to set number values for the functions based on choices. For example, when the user changes the house type in a *residential function* from a large detached dwelling to a town house, the decision table will react and change the property value of the average lot size for a house, the average water demand, the cost etc. Rules are created that calculate how many of these houses can be built in the *zone*. Similarly, rules have been developed for the other *functions* (*infrastructure function*, *park* etc.). In addition, the rule-based system is used to aggregate values of all zones, giving a summary of the total precinct.

4.2.3 The prototype

Using polygon information from GIS, *zones* can be defined. Each *zone* contains *functions* that have several properties that can be changed by the user. Each change will have an effect on the *zone* and after that on the *precinct*, which is comprised of the *zones*. The value of the properties of the *precinct* is displayed in charts. So when the user makes a change somewhere in a *zone* or in a zone function, the chart is updated automatically (Fig. 9)

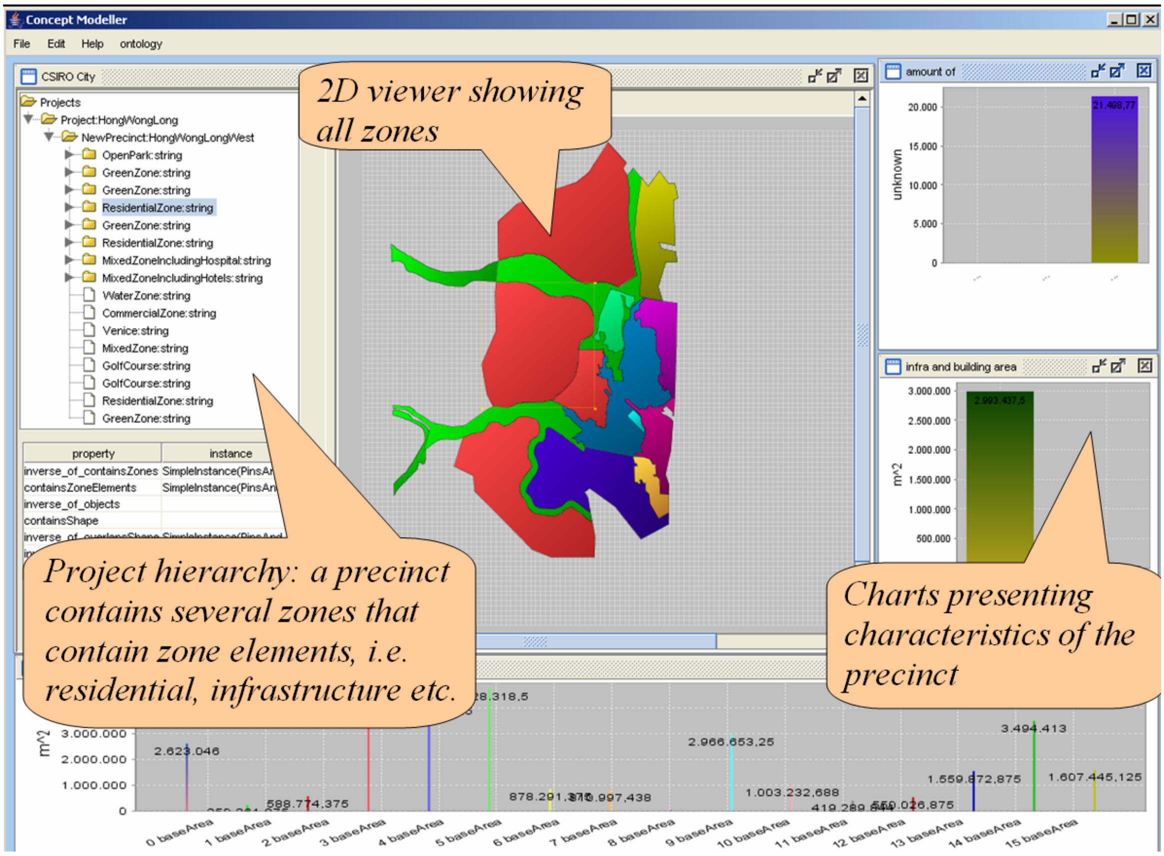

FIG. 9: A screenshot of the use of a shape model to visualise zones.

4.3 Neighbourhood prototype

4.3.1 The ontology

The neighbourhood ontology deals with urban development on a more detailed level compared to the master plan ontology. A precinct (reused from the master plan ontology) contains *elements* such as *regions, streets, parks, water, shopping malls, commercial regions* and *public transport buildings*, etc. An *element* is a subtype of *shape*, and consequently can hold geometry. *Regions* have many properties such as house type, and reuse the decision tables from the previous example containing knowledge relating to house type with lot size. Other properties are rent prices, water demand, amount of parking spaces, distances between public transport and houses, distances between houses and shopping malls, parks etc.

4.3.2 The behaviours

Several decision tables relate qualitative statements with real figures. For example, the decision table on house types resulting in a lot size is re-used. Other examples are street type information that is related to the amount of parking spaces and road capacity. The rules are mainly used to aggregate information. For example, calculation of the amount of buildings, parking spaces, roads, shopping malls, water demand, area usage, etc., is aggregated from all the elements to determine the characteristics of the total precinct.

4.3.3 The prototype

All the *site elements*, such as *shopping malls, streets, commercial regions* and *residential regions*, are visualised in the 2D viewer (Fig. 10). It is possible to select individual objects or multiple objects and change their properties such as the property ‘house type’. Changing the ‘house type’ will influence the amount of houses, the amount of people, the water demand, the density etc. These changes are directly reflected in the charts that provide an overview of the total precinct.

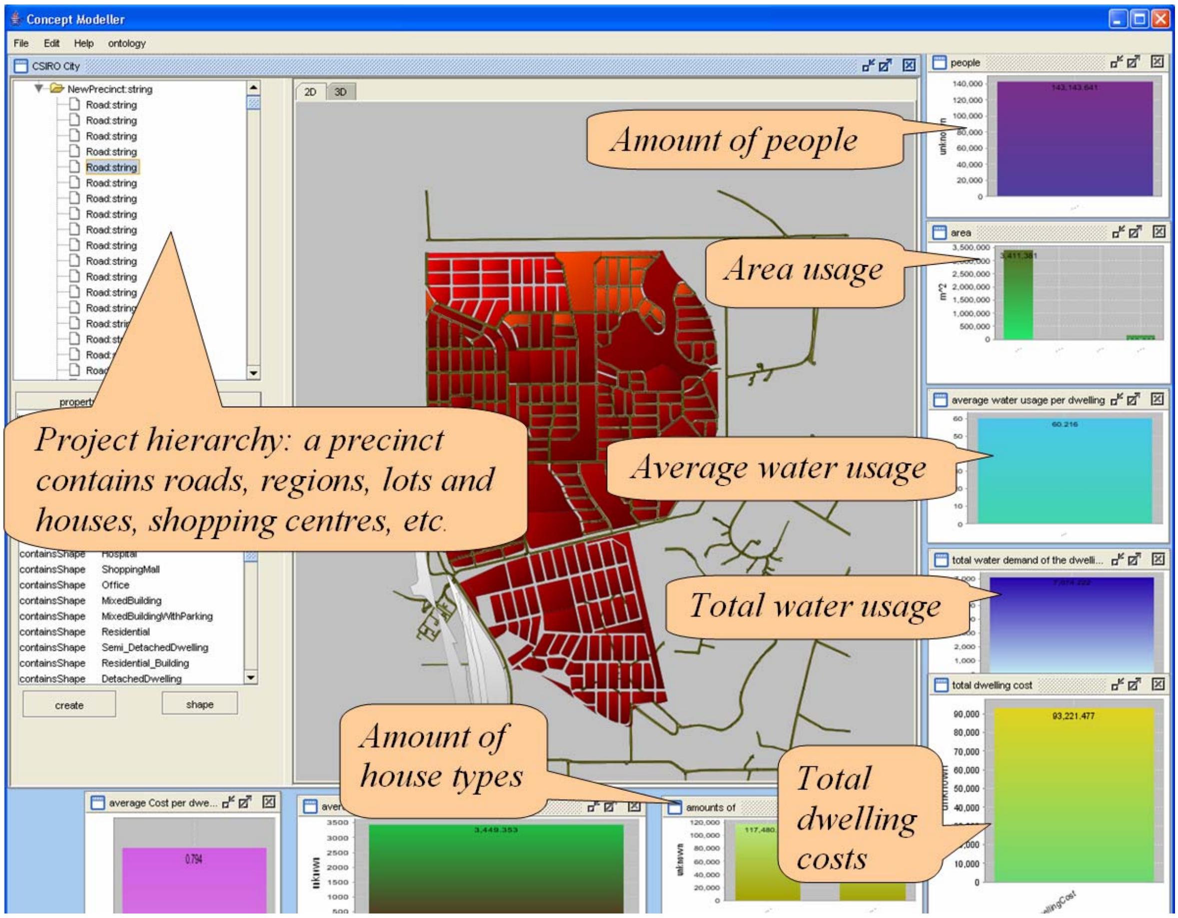

FIG. 10: A screenshot of the use of the shape model to visualise roads, lots and houses.

5. CONCLUSIONS AND DISCUSSION

The role of sustainability in urban design is becoming increasingly important, putting pressure on existing infrastructure such as water, energy and transport. Many different aspects need to be taken into account integrally in order to optimise an urban design. A framework providing interoperability between loosely coupled software applications supports the development of an integrated software system. Semantic Web offers technology to create such a framework. New applications or web services can join a network of collaborating applications and consequently extend the amount of urban development assessments.

This network of collaborating applications is possible when a network of ontologies is used. As each application can have its own disciplinary specific ontology, the different ontologies need to be aligned. This can quickly become a comprehensive task as the ontologies can span multiple domains. Inference can support this aligning task by determining class hierarchies, classifying individuals, performing consistency checks and by inferring interoperability.

Though currently some SW technologies are still under development, already a meaningful prototype application has been developed using a network of Semantic Web ontologies. The prototype is comprised of ontology driven software components which are highly re-usable. By developing an urban ontology and relating it to the existing ones, the software components can be re-used. Two prototypes have been created by developing different urban ontologies and behaviours. The behaviours can easily be inserted using different knowledge representation languages. The prototypes demonstrate the assessment of urban development on two different levels of scale.

Though the assessments are simple, the prototypes demonstrate that with the creation of a network of ontologies software and data can be re-used easily. Besides re-use of general software functionality, the reuse of urban concepts and their behaviour is also demonstrated.

6. REFERENCES

Algernon (2005). Accessed 2005 at http://www.cs.utexas.edu/users/qr/algy

AMCORD (1992). *Guidelines for urban housing*, AGPS, Canberra, Australia.

Baader F., Calvanese D., McGuinness D., Nardi D., Patel-Schneider P. (eds.) (2003). *The description logic handbook*. Cambridge University Press, UK.

Berners–Lee T., Hendler J., and Lassila O. (2001). *The semantic web*, Scientific American http://www.sciam.com. May.

Brickley D. and Guha R.V. (eds.) (2004). RDF *vocabulary description language 1.0: RDF schema*, W3C Recommendation, Accessed 2005 at http://www.w3.org/TR/rdf-schema

Chen H., Fellah S. and Bishr Y. (2005): *Rules for geospatial semantic web applications. W3C workshop on rule languages for interoperability*, Washington, D.C., USA, The World Wide Web Consortium (W3C).

CLIPS (2006). *Clips A tool for building expert systems*, Accessed January 2006 at http://www.ghg.net/clips/CLIPS.html

CRC for Construction Innovation (2005). Construction 2020: *A vision for Australia's property and construction industry*, CRC for Construction Innovation, Brisbane, Queensland, Australia.

Daconta M.C., Obrst L.J., and Smith K.T. (2003). *The semantic web: A guide to the future of XML*, Web Services, and Knowledge Management, Wiley Publishing Inc., Indianapolis, Indiana.

Fensel, D. (2002). *The semantic web: Yet another hip?*, Vrije Universiteit Amsterdam, The Netherlands.

Friedman-Hill E. (2003). *JESS in action*, Manning Publications Company, ISBN: 1930110898.

Golbreich C. and Imai A. (2004). *Combining SWRL rules and OWL ontologies with Protégé*, OWL Plugin, Jess, and Racer. 7th International Protégé Conference, Bethesda, MD.

JESS (2005*). Java expert system shell*, Accessed 2005 at http://herzberg.ca.sandia.gov/jess/

Knublauch, H. (2003) An AI tool for the real world: Knowledge modeling with Protege. JavaWorld, available at http://www.javaworld.com/javaworld/jw-06-2003/jw-0620-protege.html

Lieberman J., Pehle T. and Dean M. (2005): *Semantic evolution of geospatial web services*. W3C workshop on frameworks for semantics in web services, Innsbruck, Austria.

Map Bureau (2003). *RDFMap*, Accessed 2005 at http://www.mapbureau.com/rdfmap1.0/

MathML (2002). Accessed November 2005 at http://www.w3.org/Math

Motta E. (1999). Reusable Components for Knowledge Models. IOS Press, Amsterdam.

OWL-S (2005). Accessed 2005 at http://www.daml.org/services/owl-s/1.0/

PELLET, Accessed 2005 at http://www.mindswap.org/2003/pellet/index.shtml

Protégé (2005). Stanford Medical Informatics, Accessed 2005 at http://protege.stanford.edu

Prolog (2005), Accessed 2005 at http://gnuprologjava.sourceforge.net/

RACER, Accessed 2005 at http://www.racer-systems.com/

Schevers H. (2004). *Demand support by virtual experts. Supporting the client during the inception phase of a building and construction project*. PhD thesis 2004. ISBN 90-9019215-2, Schijndel, the Netherlands.

Smith M.K., Welty C. and McGuinness D. (2004). *OWL web ontology language guide*, Accessed 2005 at http://www.w3.org/TR/owl-guide/

SOAP (2005). Accessed 2005 at http://www.w3.org/TR/SOAP/

Sowa J.F. (2000). *Knowledge representation: Logical, philosophical, and computational foundations*, Brooks/Cole Publishing Co., Pacific Grove, CA, 2000.

SWRL (2005). Accessed 2005 at http://www.w3.org/2003/11/swrlb.owl

UDDI (2005). Accessed 2005 at http://www.uddi.org/

Web Ontology Language (2004). Accessed 2005 at http://www.w3.org/2004/OWL/

WSDL (2005). Under world wide web consortium. Accessed December 2005 at http://www.w3.org/TR/wsdl

WSMO (2005). Web service modelling ontology, Accessed December 2005 at http://www.wsmo.org/