OntoWeb Travelling Domain Experiment

Results for *Open*KnoME

Dr Jeremy Rogers
Medical Informatics Group
University of Manchester
United Kingdom
jeremy@opengalen.org

Introduction

OpenKnoME [1] is a client software environment for :

- primary authoring of GRAIL ontologies in a collaborative, distributed multi-author setting
- linking GRAIL ontologies to external data sources
- rapid prototyping of subdomain and task specific ontological schemas (intermediate representations) that are systematic simplifications of a more complex, shared, common ontology [2]

*Open*KnoME, written in Visualworks Smalltalk, uses GRAIL formalism reasoners, which are instantiated as separate server applications. The direct descendent of the original GRAIL development and prototyping environment, *Open*KnoME is the product of more than a decade's research into large scale collaborative ontology building, maintenance and delivery, centred on what is now the *Open*GALEN Common Reference Model of medicine (CRM). *Open*KnoME is available under open source license for non-commercial use only from www.topthing.com.

GRAIL is a relatively old formalism , related to more modern and mainstream description logics. It includes many common DL constructors including existential role restriction, role hierarchies and role transitivity. The language is declarative, and compilation is statement order dependent. The compiled model, however, contains relationships between concepts that were not explicitly stated in the sources but were inferred during compilation.

GRAIL does not support true negation or disjunction, has no notion of instances, no mechanism to declare siblings as either truly disjoint or as covering the domain, and only a very limited implementation of cardinality. GRAIL does not provide native support for specific data types (such as dates, or numerical ranges).

Additionally, however, GRAIL *does* include a role inheritance constructor - also known as refinement or specialisation – that is not supported by any current description logic.

GRAIL models, like any compositional ontology, are inherently dynamic: they are typically constructed as a collection of separate mini-ontologies that can be combined, subject to permissions and constraints also in the model, to build detailed concepts that form the target ontology. However, the terms in the target ontology can not be exhaustively defined explicitly, but instead are an implied conceptual space. It is neither sensible nor possible to pre-enumerate all possible members of the implied ontological space because the number of possible constructs in that space rapidly rises to many billions. For this reason a static dump of a 'fully populated' travel model is not provided as part of this experiment.

Building the Model

Generic Model

The OpenGALEN generic model (an upper ontology) [3] was used as the starting point. This ontology provides an off-the-shelf re-usable, relatively domain independent, upper ontology comprising structures, process and substance together with a rich library of semantic links and associated coherent transit and GRAIL role inheritance rules. This upper ontology was developed as a component of the *Open*GALEN CRM ontology. For this evaluation the upper ontology was pruned slightly to remove a small surviving residual of more clinically oriented content.

A new source file project manager was created within the OpenKnoME, and a copy of the OpenGALEN generic model sources was imported.

New Category Space

Primary knowledge acquisition involved reading the textual scenario description to extract candidates for new elementary classes and semantic link types.

Modes of transport

An ontology of vehicles was constructed, primary classification being by whether the vehicle travels on land, water or through the air. I noted that the scenario description stated that no other forms of transport existed other than planes, trains, cars, ferries, motorbikes and ships. However, the same scenario description later mentions buses and underground transportation. A more detailed and general ontology of vehicles was therefore sketched. Additional and atypical forms of vehicle were considered, such as amphibious vehicles, seaplanes, flying cars, gliders, hot air balloons, water taxis, helicopters, trams, bicycles and rickshaws. These implied further classification of vehicles by mode of propulsion.

The next step was to attempt to construct a common parent of all modes of transport. For this, other non-vehicular forms of transportation were also considered, such as horses and camels. From this it became clear that the concept ‘mode of transport’ would necessarily be a fairly abstract notion, subsuming some built structures (vehicles) as well as some organisms.

The concept of the process of transporting was modelled. It may be further described by the physical mode of transport used, by subprocesses of arriving and departing (each of which may be further described by a time and a location), and by a goal (for example, visiting a specific attraction or a city). The more abstract idea of a travel itinerary, comprising any number of transports and hirings of rooms was next modelled. No attempt was made to model any temporal ordering of the various journeys and hotel bookings, as temporal reasoning is not directly supported within GRAIL.

A list of companies was declared, and the processes of hiring or manufacturing created. Airliners were permitted to be characterised by who made them, and independently to take any model type designation. Constraints were entered to say that any airliner with an specific model type designation must have been manufactured by the appropriate manufacturer, and that any airliner made by a specific manufacturer can only have a model type appropriate to that manufacturer.

Trade and Commerce

A small model of trade and commerce was built, in which relationship are declared between [Hiring] as an act, [Price] and [Currency].

Accomodation

Hotels were modelled to have rooms as discrete components. Rooms have a number of further properties (whether they contain a TV, minibar etc), and can be hired from a specified set of companies. Hotels themselves have a set of characteristics that would normally be populated by free text or numerical data. These are included in this model by way of illustration only. A more appropriate mechanism to store instance information (physical address, URL etc) would be in an external database. The role of ontologies in indexing actual products has previously been explored using this toolset in the UK Drug Product Ontology project.

Geography

A list of cities, continents, countries and locations (museums, airport, beaches) was declared as primitives. The inclusion of apparent instances such as 'Paris' and 'Tacoma Airport' within a framework that does not support instances is explained by the fact that, within this model, the identifier 'Paris' represents that class of all possible Paris's, of which subclasses might include 'Paris on Bastille Day', 'Paris in the Spring'.

The partitive relationships between these geographic loci were modelled using standard Winston-Odell partonomic relationships, which are provided in the upper ontology.

Pragmatics of travel

The scenario asked that we attempt to model some real world constraints, such as that it is not possible to cross the Atlantic by car, or that while it might be possible to travel from Vladivostok to Johanesburg by car, nobody would want to. A very small subset of all the constraints that might be required to constrain the model to allow only physically plausible modes of travel between specified locations were expressed for illustrative purposes only. The constraints are in the form of 'a journey between locations of a particular type can only be made using specified modes of transport', and one result is that an attempt to form the concept of a journey from New York to Cairo by bus will fail.

Examples:

The class of trips, of which John's would be an instance, is expressed as a single expression:

```
TravelItinerary which <
        hasStructuralComponent (Flying which <
                hasSpecificSubprocess (Arriving which <
                        hasSpecificLocation JFK
                        occursDuring April05>)
                hasSpecificSubprocess (Departing which <
                        hasSpecificLocation Madrid
                        occursDuring April05>)
                hasGoal (Visiting which actsOn StatueOfLiberty)>)
        hasStructuralComponent (Flying which <
                hasSpecificSubprocess (Arriving which <
```

```
                hasSpecificLocation DullesAirport
                occursDuring April11>)
        hasSpecificSubprocess (Departing which <
                hasSpecificLocation Madrid
                occursDuring April11>)>)
hasStructuralComponent (Flying which <
        hasSpecificSubprocess (Arriving which <
                hasSpecificLocation Madrid
                occursDuring April15>)
        hasSpecificSubprocess (Departing which <
                hasSpecificLocation DullesAirport
                occursDuring April15>)>)
hasStructuralComponent (Hiring which <
        hasSpecificPersonPerforming HolidayInn
        actsSpecificallyOn (Room which isStructuralComponentOf
                (Hotel which hasSpecificLocation JFK))>)
hasStructuralComponent (Hiring which <
        hasSpecificPersonPerforming HolidayInn
        actsSpecificallyOn (Room which isStructuralComponentOf
                (Hotel which hasSpecificLocation DullesAirport))>)>
```

..and this is classified automatically under TransatlanticTrip and 'TripToVisitNewYork', even though the USA locations are only identified as airports or monuments.

The complex graph description (above) can be entered as a single expression, within a normal ASCII text editor. It is not a requirement that it be assembled from smaller subgraphs that are declared, evaluated and named separately *a priori* before the larger composition can be expressed.

User Interface

The OpenKnoME includes prototype tools to dynamically construct structured data entry interfaces based on the ontology. A screen shot demonstrates a form generated on the topic of a trip, or transport event, partially completed and showing the sub-form produced in response by the user to describe the departure in more detail.

References

1. Rogers J.E., Roberts A., Solomon W.D., van der Haring E, Wroe C.J., Zanstra P.E., Rector, A.L. (2001) GALEN Ten Years On: Tasks and Supporting tools Proceedings of MEDINFO2001, V. Patel et al. (Eds) IOS Press;: 256-260
2. Solomon W.D., Wroe C.J., Rector, A.L., Rogers J.E., Fistein J.L., Johnson P. (1999) A Reference Terminology for Drugs Annual Fall Symposium of American Medical Informatics Association, Washington DC. Hanley & Belfus Inc. Philadelphia PA;:152-155
3. www.opengalen.org/open/crm