Travelling Domain Experiment: Engineering with OntoEdit

York Sure
Institute AIFB
University of Karlsruhe
76128 Karlsruhe, Germany
sure@aifb.uni-karlsruhe.de

1 Introduction

This paper presents the results of using **OntoEdit** in the context of the experiment on evaluation of ontology related technologies that was initiated by the Special Interest Group (SIG) on Enterprise-Standard Ontology Tools of the EU IST-2000-29243 thematic network OntoWeb (cf. http://www.ontoweb.org/).

OntoEdit [1,2] is a collaborative ontology engineering environment that has been developed keeping five main objectives in mind: (i) Ease of use. (ii) Methodology-guided [3] development of ontologies. (iii) Ontology development with help of inferencing. (iv) Development of ontology axioms. (v) Extensibility through plugin structure [4].

Modelling ontologies using OntoEdit involves modelling at a conceptual level, viz. as independently of a concrete representation language as possible, and using GUI's representing views on conceptual structures (concepts, concept hierarchy, relations, instances, axioms) rather than codifying conceptual structures in ASCII. The conceptual model of an ontology is stored internally using a powerful ontology model, which can be mapped onto different, concrete representation languages (e.g. RDF(S) or DAML+OIL). As mentioned above, the core functionalities of OntoEdit are easily expandable through a flexible plug-in framework.

In order to provide a clearly defined semantics to the knowledge model of OntoEdit, the knowledge structures of OntoEdit correspond to a well-understood logical framework, viz. F-Logic [5] ("F" stands for "Frames"). F-Logic allows for concise definitions with object oriented-like primitives (classes, attributes, OO-style relations, instances) that fit very nicely with the OntoEdit GUI. Furthermore, it also has PL-1 like primitives (predicates, function symbols). Furthermore, F-Logic allows for axioms that further constrain the interpretation of the model. Axioms may either be used to describe constraints or they may define rules, e.g. in order to define a relation R by the composition of two other relations S and Q. F-Logic rules have the expressive power of Horn-Logic with negation and may be transformed into Horn-Logic rules. Unlike Description Logics (DL), F-Logic does not provide means for subsumption, but (also unlike DL) it provides for efficient reasoning with instances and for the capability to express arbitrary powerful rules, e.g. ones that quantify over the set of classes. Cf. [2] for a more elaborated discussion.

Our inference engine Ontobroker [6] comes with several features that makes it adequate as a backbone for an ontology editor. In particular, it provides: (i) A namespace mechanism: Thus, several ontologies (or ontology parts) may be syntactically split into modules and processed by different inference engines. (ii) Switch-off: It is possible to switch of (possibly singleton) sets of definitions. Thus, one may test interactions and easily distinguish between modules. (iii) DB Connectors: Thus, one may easily map database tables into predicates via JDBC. (iv) User-definable built-Ins: Besides of standard built-ins like "multiply", the user may

define his own ones for special purposes. (v) An extensive API: Thus, one may remotely connect to the inference engine and one may also import and export several standards (e.g., RDF(S)).

2 Engineering the Model

For engineering the traveling domain model with OntoEdit we performed the two steps "Kickoff" and "Refinement" of our methodology.

2.1 Kickoff

In the first step, a semi-formal description of the ontology is created by sketching the most relevant elements of the domain. The early stages of ontology development are often driven by brainstorming like knowledge acquisition sessions. In other projects (cf., e.g., [7]) we made good experiences with creating mindmaps as a first draft of relevant elements for a domain. Especially domain experts who were not familiar with modeling preferred using a mindmapping tool instead of directly modeling with an ontology editor. Figure 1 shows the mindmap created from the natural language description of the domain. We rely on a commercial tool for the creation of electronically mindmaps, the MindManager 2002 Business Edition (cf. http://www.mindjet.com/).

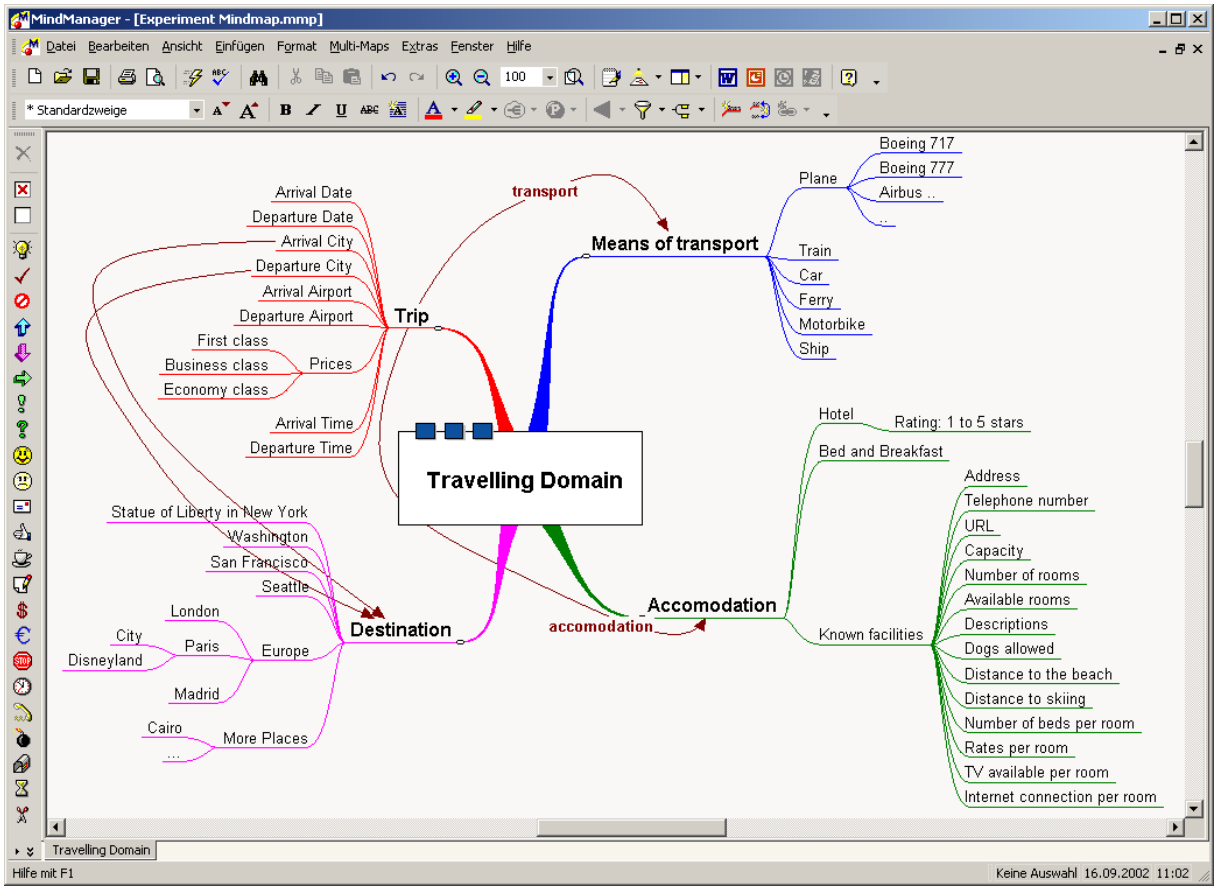

Fig. 1: MindMap of the traveling domain

When collaborating with domain experts the time needed for knowledge acquisition is essential, especially in industrial environments. The advantage of using mindmaps is (i) the quick generation of a graphical representation of relevant domain elements (ii) by using an intuitive and rather well-known tool. The creation of this mindmap took less than 20 minutes.

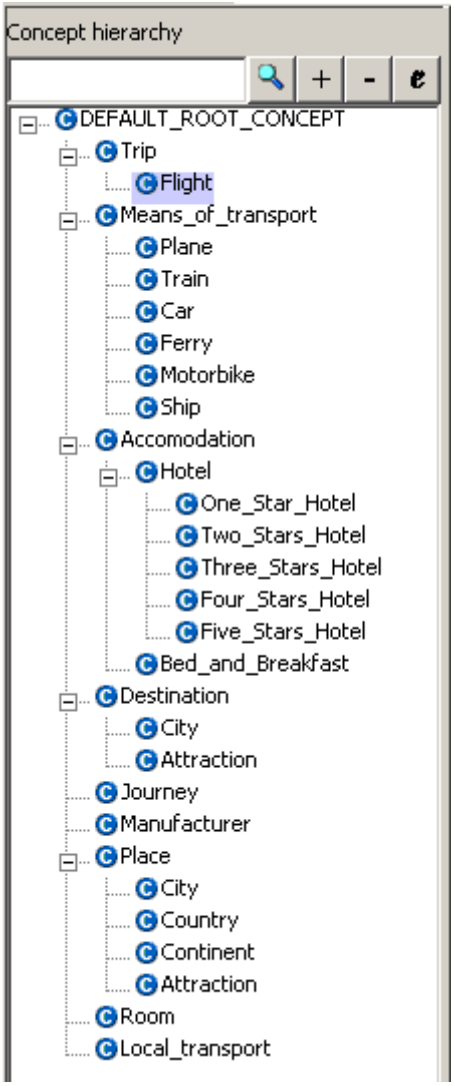

Fig. 2: Concepts

However, when it comes to terms of a formal model of the domain, this representation is no longer suitable. This representation does not clearly distinguish between the notions of concepts, relations etc.. The only semantics for connections (branches or directed edges) in a mindmap is that these elements are "associatively linked". Closer related elements are typically marked with same colors. Typically a mindmap represents the key concepts and their relationships and to formalize it into an ontology, the ontology engineer has now to decide which elements are concepts, how is their hierarchical "is-a" structure and which elements are other named relationships. In some cases one might find prototypical instances, but constraints like the ones given at the end of the domain description are typically not found in mindmaps.

Currently the formalization has still to be "sorted out" manually by ontology engineers. A XML based exchange between the MindManager and OntoEdit guarantees interoperability on a syntactical level. For future versions we plan also to support the decision making during the formalization.

2.2 Refinement

2.2.1 Concepts and relationships

To formalize the mindmap we followed the steps (i) creation of an "is-a" hierarchy of concepts, (ii) adding attributes of concepts and relationships between concepts other then "is-a", (iii) including prototypical instances and (iv) adding axioms that represent constraints and common sense deductions. At several points we needed to introduce further concepts, that were not obvious at first hand from the domain description but necessary to build a complete model. E.g. we introduced a concept `Journey` to combine several trips, some others are mentioned in the text below. This reflects the fact that the mindmap is typically covering only the most relevant elements of a domain, but is not intended to represent more complex relationships in a formally consistent way.

Figure 2 shows the resulting concept hierarchy. When defining a `Trip` we made the following assumption to narrow down multiple possible interpretations in the description: Not only for flights, but for every `Trip` the arrival/departure date, arrival/departure city is known. A `Flight` is a specialization of `Trip` for which additionally the arrival/ departure airport and the prices for first/business/economy class are known. The grey shaded relationships shown in Figure 3 illustrate the inherited relationships for the selected concept `Flight`, i.e. the domain of them is `Trip`. The relationships without shading have as a domain `Flight` itself.

We made some simple assumptions for defining ranges of the relationships: e.g. the dates are coded as `STRING` which directly points to the XML Schema definition for strings (http://www.w3.org/2001/XMLSchema#String) – same holds e.g. for prices/`INTEGER`.

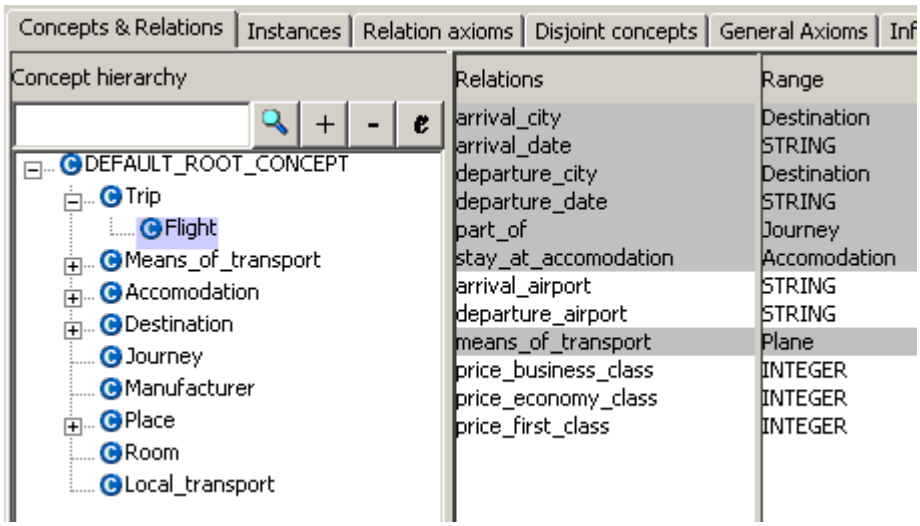

Fig. 3: Relationships of „Flight“

The relationship `means_of_transport` is firstly defined for `Flight` with the range `Means_of_transport`. We then refined it for `Flight` by specializing the range to `Plane` that is a subconcept of `Means_of_transport` (cf. Figure 4).

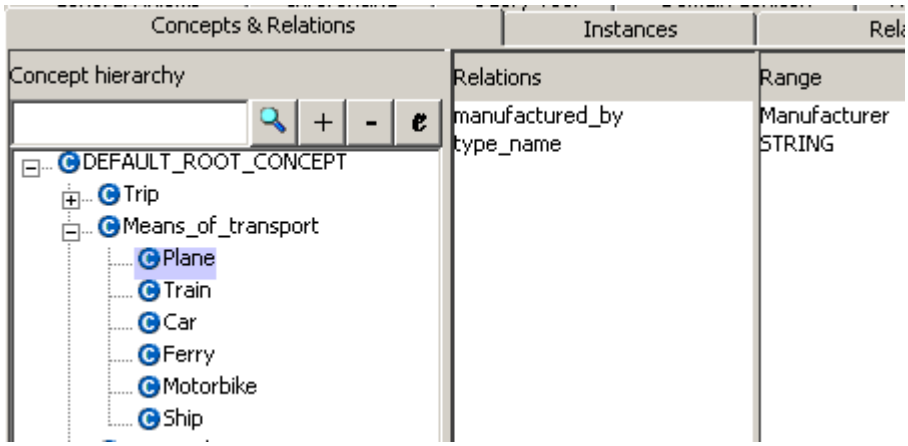

Fig. 4: Relationships for „Plane“

`Accomodation` (cf. Figure 5) has subconcepts `Hotel` and `Bed and Breakfast`. Beside relationships for the facilities mentioned in the description (address, available rooms etc. ..) one can see relationships to `Room` and `City`. To model the star ranking schema for hotels we added further specializations of `Hotel`, e.g. `One-Star-Hotel` (see later in the subsection about instances how we model a particular hotel as an instance). To model that each hotel belongs to one star category, we defined `Hotel` as an “abstract” concept, i.e. there are no instances of this concept allowed, and all subconcepts like `One Star Hotel` as “concrete” concepts (cf. Figure 6). Same holds e.g. for `Means of transport` and its subconcepts.

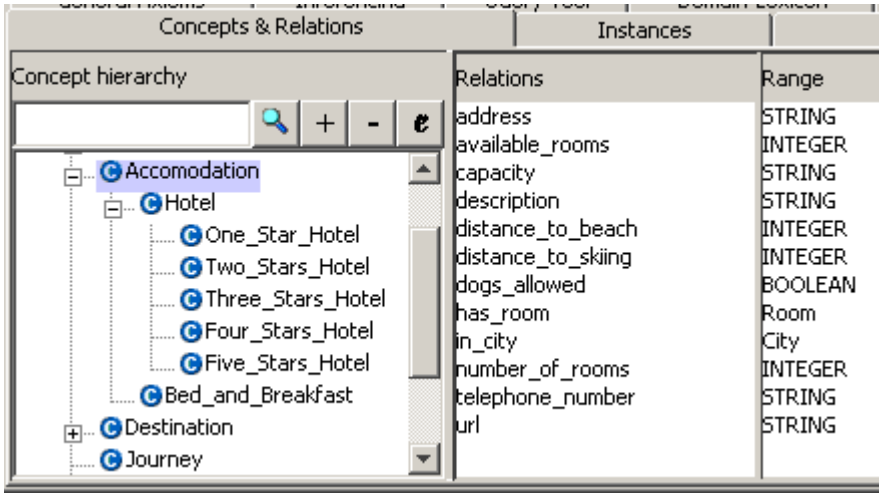

Fig. 5.: Relationships of „Accomodation“

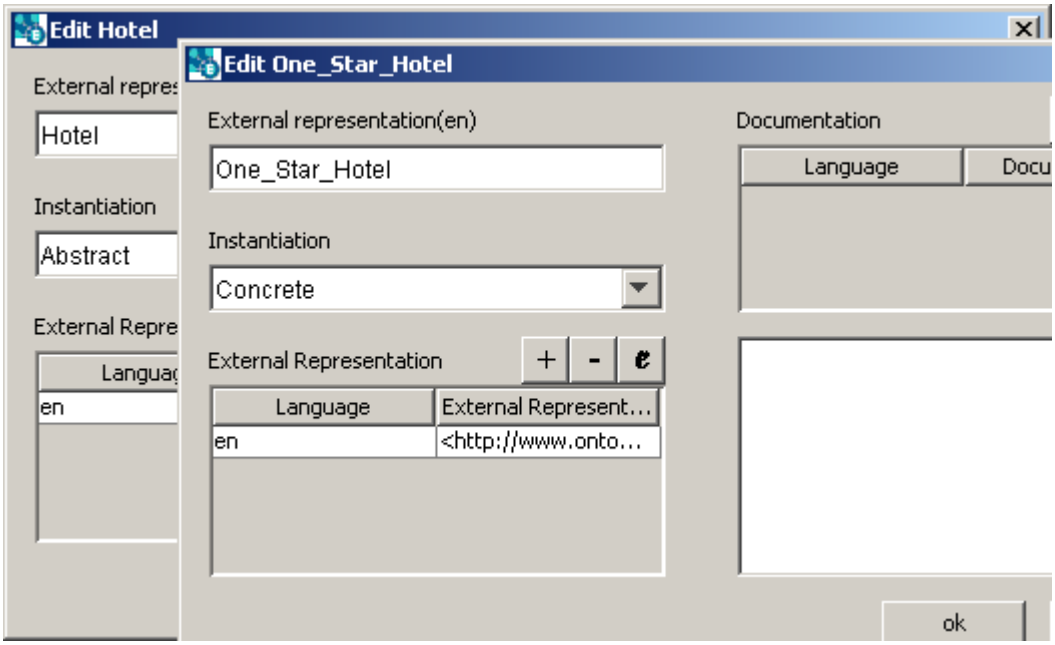

Fig. 6: “Abstract” vs. “concrete” concepts

We introduced `Place` as a superconcept of `City` (cf. Figure 7). For further axioms on top we also included `Attraction`, `Country` and `Continent`. The concepts are related via the `located in` relationship, e.g. an `Attraction` is `located in` a `City`, a `City` is `located in` a `Country` and a `Country` is located in a `Continent`. As shown later this relationship is transitive. `City` and `Attraction` are also subconcepts of `Destination`, i.e. they are multiply inherited. Alternatively one could consider to add `Destination` as a subconcept of `Place`, too. In the current scenario that would have no effect. Modeling it this way seemed more intuitive to us.

Last but not least, a `Journey` has potentially many parts, i.e. it can be related via `has part` to many instances of `Trip` that belong to this `Journey`. For completeness we included also the inverse relationship part of for `Trip` with the range `Journey` and defined these two relationships as invers (see later subsection on axioms).

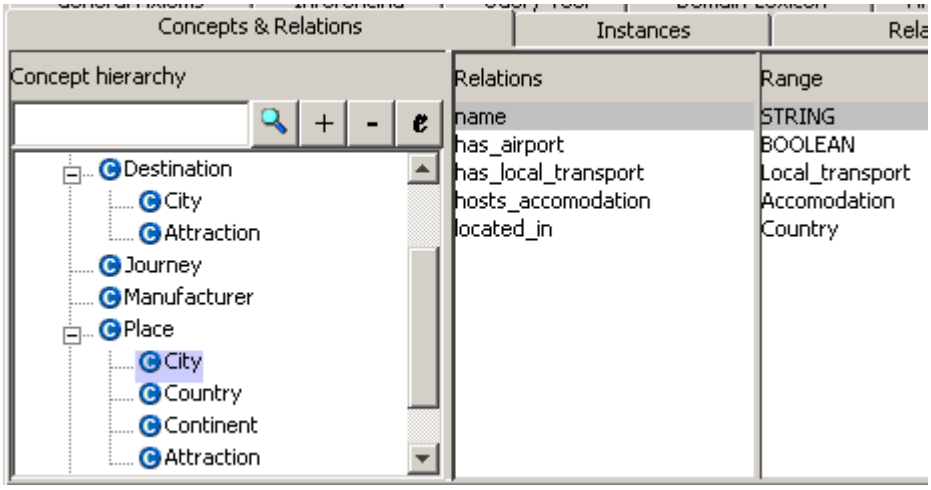

Fig. 7 : Relationships of "City"

2.2.2 Instances

We modeled several instances, e.g. shown in Figures 8, 9 and 10. Part of them are given in the first part of the description (e.g. the cities `New York`, `Washington` etc. and the attractions `Statue of Liberty` and `EuroDisney`) , others are given in the last section with an example journey for John (cf. Figure 10). John makes two flights (from `Madrid to NY` and from `Washington to Madrid`) and one trip with a motorcycle (from `NY to Washington`).

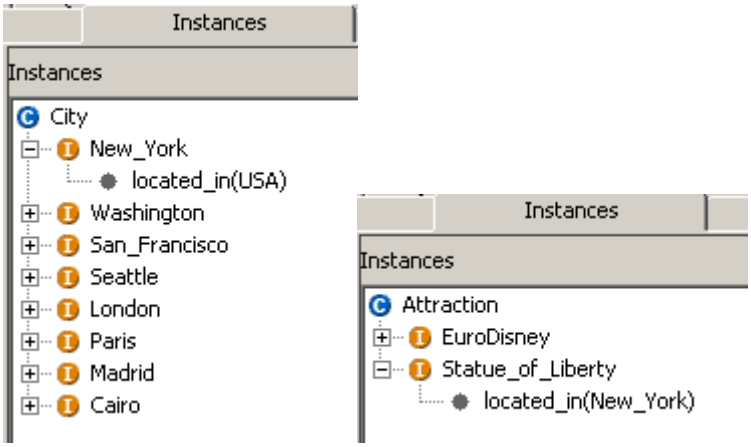

Fig. 8 : Instances of "City" and "Attraction"

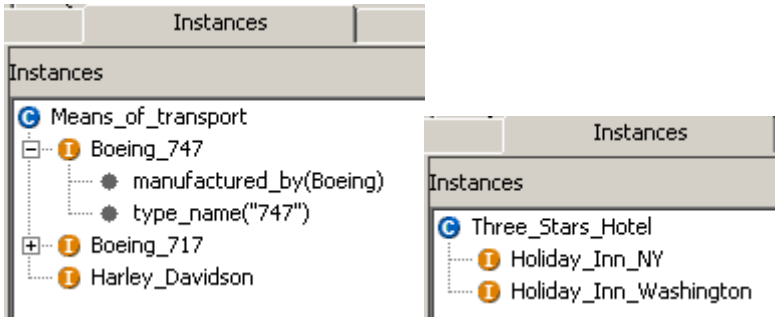

Fig. 9: Instances of "Means of transport" and "Three Stars Hotel"

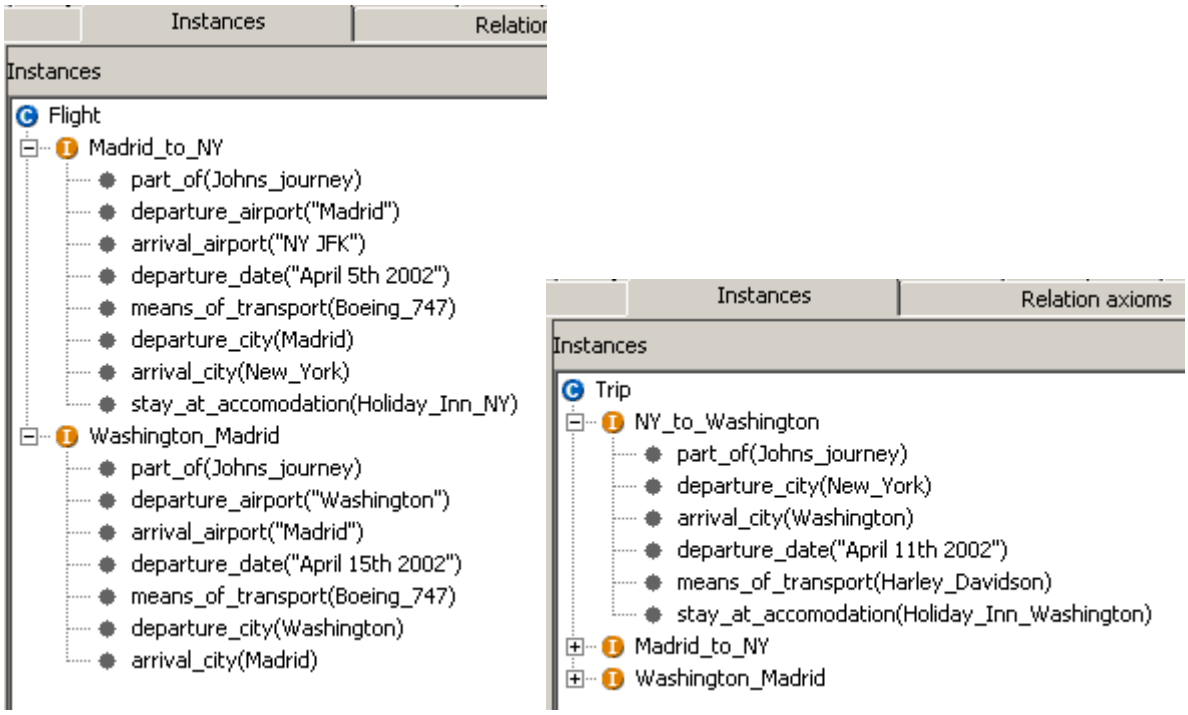

Fig. 10: Instances of "Flight" and "Trip"

2.2.3 Axioms

On top we defined several axioms: `located in` is transitive (cf. Figure 11), e.g. `has part` is inverse to `part of` (cf. Figure 12), the subconcepts of `Hotel` are pairwise disjoint (cf. Figure 13).

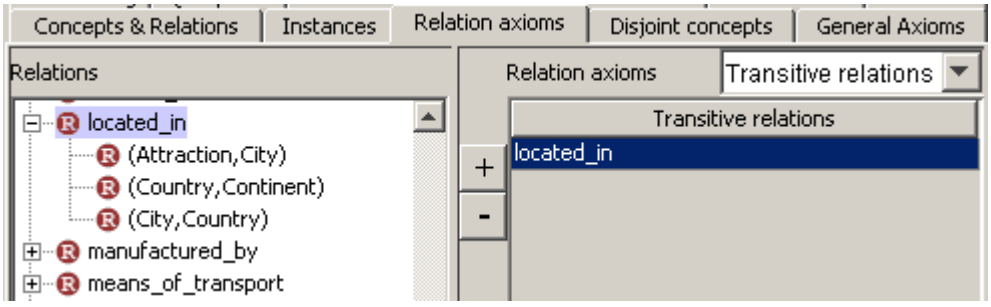

Fig. 11: Transitive relationship

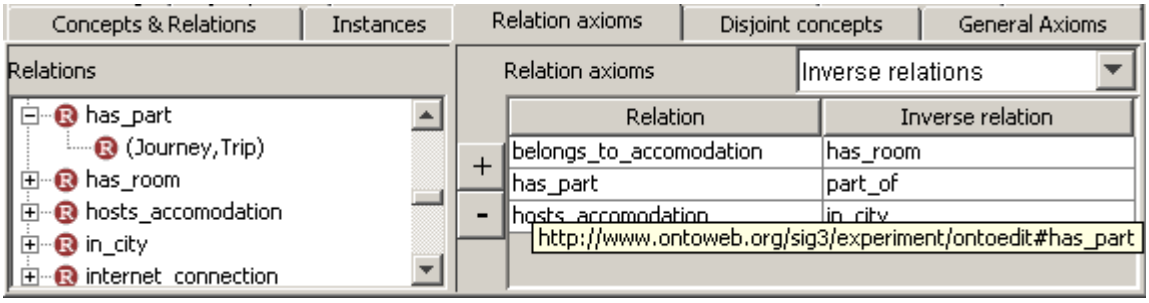

Fig. 12: Inverse relationships

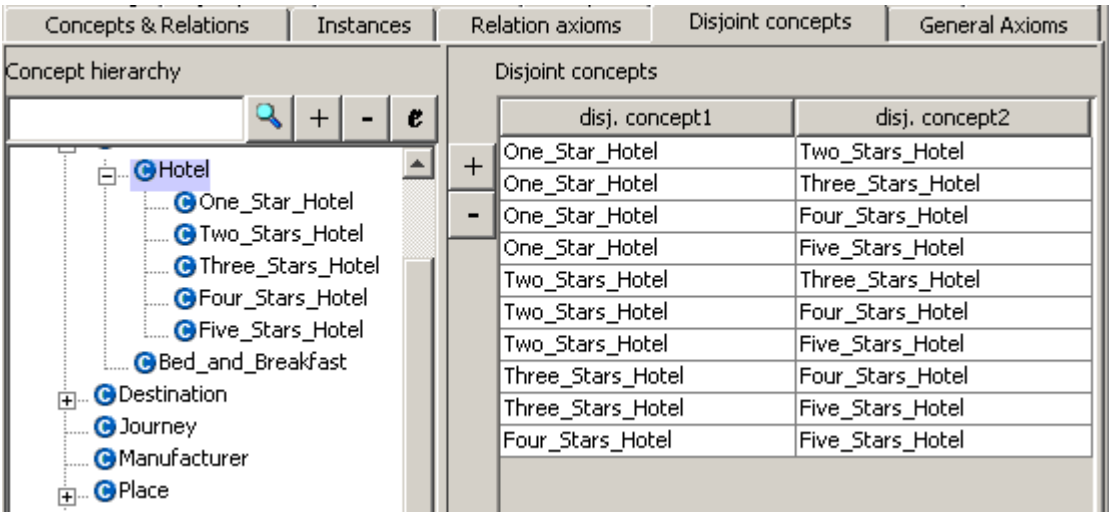

Fig. 13: Disjoint concepts

A more complex axiom is given in the description by "it is not possible to go from America to Europe by train, car, bike or motorbike" (without restricting the generality we excluded bike because it was not given in the previous section for means of transport). We defined a general axiom in F-Logic that can be used to check whether this constraint holds for all given instances (see also Figure 14):

```
FORALL T check("You cannot travel from North-America to Europe
by train, car or motorbike!",T)
     <- EXISTS M,D,A
     T:Trip[departure_city->>D;
            arrival_city->>A;
            means_of_transport->>M]
     AND D:City[located_in->>"North_America"]
     AND A:City[located_in->>"Europe"]
     AND (M:Train OR M:Car OR M:Motorbike).
```

Other given constraints can be formalized similar to this. To perform a check we simply query for all values of the 2-ary predicate `check`.

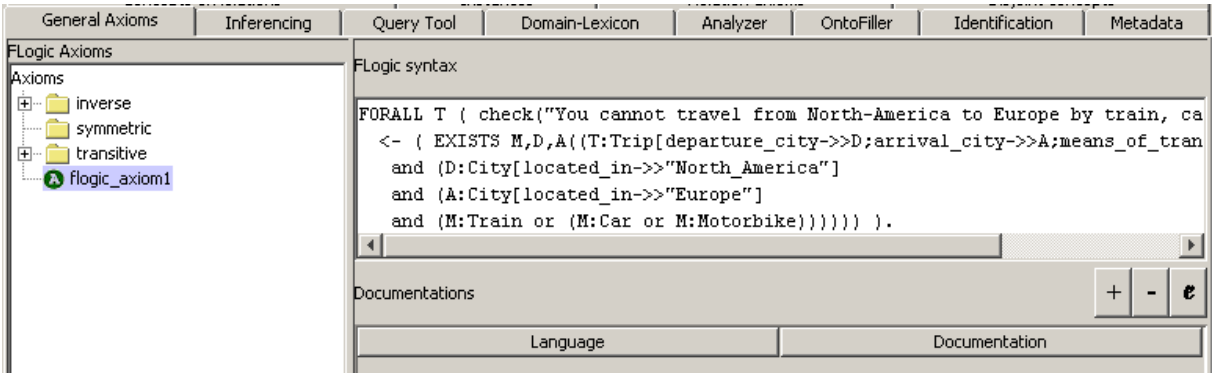

Fig. 14: General axiom

2.2.4 Inferencing

OntoEdit (Inferencing Edition) can be connected to the inferenced engine Ontobroker. We are thereby able to perform queries for concepts, relationships, instances etc.. E.g. we can ask for all cities and where they are `located in`. If we enable the axiom for transitivity of the relationship `located in` (like shown in Figure 15) we receive as an answer to that query that e.g. `New York` is located in `USA` (an instance of `Country`) as well as the fact the `New York` is located in `North America` (an instance of `Continent`).

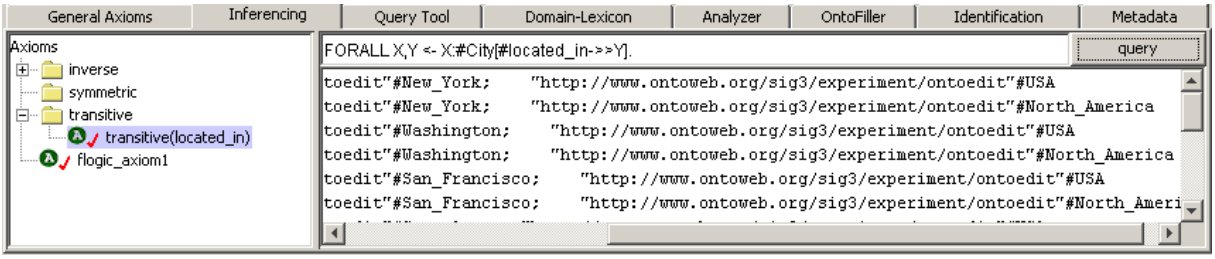

Fig. 15: Inferencing in OntoEdit

3 Conclusion

We illustrated the modeling process for engineering the traveling domain with OntoEdit. We were able to formalize the given natural language description. Our aim was to model the domain as close as possible to the given domain description, i.e. we tried to add only those concepts and relationships that were mentioned. On top we defined axioms that reflect constraints given in the description.

4 References

[1] Y. Sure, M. Erdmann, J. Angele, S. Staab, R. Studer and D. Wenke. **OntoEdit: Collaborative Ontology Engineering for the Semantic Web.** In: *Proceedings of the first International Semantic Web Conference 2002* (ISWC 2002), June 9-12 2002, Sardinia, Italia, Springer, LNCS 2342, pages 221-235.

[2] Y. Sure, S. Staab, J. Angele. **OntoEdit: Guiding Ontology Development by Methodology and Inferencing.** In: *Proceedings of the International Conference on Ontologies, Databases and Applications of SEmantics* (ODBASE 2002), October 28 - November 1, 2002, University of California, Irvine, USA, Springer, LNCS.

[3] S. Staab, H.-P. Schnurr, R. Studer, and Y. Sure: **Knowledge Processes and Ontologies.** In: *IEEE Intelligent Systems* 16(1), January/Febrary 2001, Special Issue on Knowledge Management.

[4] Siegfried Handschuh. **Ontoplugins –a flexible component framework.** Technical report, University of Karlsruhe, May 2001.

[5] M. Kifer, G. Lausen, and J. Wu. **Logical foundations of object-oriented and frame-based languages.** *Journal of the ACM*, 42:741–843, 1995.

[6] S. Decker, M. Erdmann, D. Fensel, and R. Studer. **Ontobroker: Ontology based access to distributed and semi-structured information.** In: R. Meersman et al., editor, *Database Semantics: Semantic Issues in Multimedia Systems*. Kluwer Academic, 1999.

[7] Th. Lau and Y. Sure. **Introducing Ontology-based Skills Management at a large Insurance Company.** In: *Proceedings of the Modellierung 2002, Modellierung in der Praxis - Modellierung für die Praxis*, Tutzing, Deutschland, 25.-27. März 2002.