Extending REA Models with a Reference Model for Abstraction Mechanisms

Wim Laurier, Geert Poels

Department of Management Information and Operational Management, Faculty of Economics and Business Administration, Ghent University, Tweekerkenstraat 2, 9000 Ghent, Belgium
{Wim.Laurier,Geert.Poels}@UGent.be

Abstract. This paper proposes a pattern for the main categories of abstraction mechanisms used in conceptual modelling. This pattern builds upon Geerts' and McCarthy's analysis of policy-level specifications, adding a conceptual relations ship (i.e. extension) that is not explicitly addressed by them although it is contained in the unified foundational ontology (i.e. UFO). As this reference model is an extension to Geerts and McCarthy's work on policy-specifications, it is ideal for extending the REA reference model with abstraction mechanisms.

Keywords: Typification, Grouping, Part-Whole, Resource-Event-Agent

1 Introduction

Geerts and McCarthy [1] advance that *"Policy-level specifications define constraints and guidelines under which an enterprise operates, and they are an extension to the REA enterprise model, adding the "what should, could, or must be" to the "what is."* In this paper, we advocate that the basic concepts used for such policy-level specifications can be positioned within a larger framework of abstraction mechanisms, which is based in a foundational ontology and has to support modelers in creating models with explicit semantics that are based in a shared conceptualisation (cf. *"an ontology is a formal specification of a shared conceptualisation."*[2]).

This framework, which we refer to as the *abstraction pattern*, captures common abstraction mechanisms that are used in conceptual modelling (e.g. aggregation and composition, generalization and specialization, classification) along with less used set-related abstraction mechanisms (e.g. membership or grouping). The abstraction pattern has its foundation in the Unified Foundational Ontology (UFO) [3], which is an upper-level ontology specifically crafted for providing an ontological foundation to conceptual modelling.

Section 2 presents the abstraction pattern. Section 3 compares it to related conceptual modelling research and practice. Section 4 contains an exhibit for the application of the pattern. Conclusions and ideas for further research are presented in section 5.

2 Abstraction Pattern

In UFO, a distinction is made between three elementary conceptual modelling categories, i.e. types, groups and individuals. **Types** are called entity types in UFO and are represented as entity types in Entity-Relationship diagrams [4] and classes in UML class diagrams. In UFO an entity type is defined as *"an entity that has an extension (being a set of entities that are instances of it) and an intension, which includes an applicability criterion for determining if an entity is an instance of it" [3]*. Potential attributes for the TYPE class (fig. 1) are those characteristics shared by the type's instances that are used in the applicability criterion that defines the entity type's intension (e.g. indications of allowed size, color, weight; identification criteria like 'a car has wheels'). **Individuals** are *"entities that are not entity types"* [3], meaning that they are instances of entity types that have no instances of their own. Potential attributes for the INDIVIDUAL class (fig. 1) point towards a unique identity (e.g. chassis number, social security number) **Groups** map to sets, which are *"entities that have other entities as members"* [3] because they represent collections of individuals. Potential attributes for the GROUP class (fig. 1) identify the characteristics that are shared by the members (e.g. the group al all red things share the same color) or characteristics of the group itself (e.g. the number of members and their average or summed weight and size).

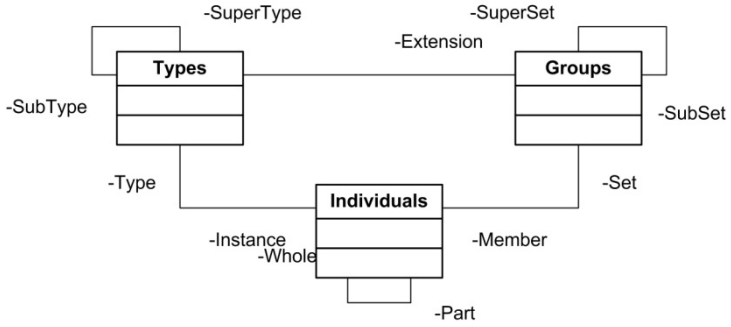

Fig. 1. Abstraction Pattern

Fig. 1 shows six kinds of abstraction relationships. First, part-whole relationships are represented by the recursive relationship on the INDIVIDUAL class. Part-whole relationships assign PART and WHOLE roles to INDIVIDUAL objects and signify that in the real world one INDIVIDUAL object is part of another INDIVIDUAL object, which is the whole that the first INDIVIDUAL object is part of.

Second, subtype-supertype relationships are represented in Fig. 1 by the recursive relationship on the TYPE class. These relationships express specialization and generalization semantics, meaning that a supertype (identified by the SUPERTYPE role) generalizes different subtypes by emphasizing the characteristics that are shared by the instances of these subtypes and a subtype (identified by the SUBTYPE role) specializes a supertype by describing additional characteristics only exhibited by the instances of the subtype considered. Subtype-supertype relationships are also known as 'is-a' relationships as each instance of a subtype is also an instance of the supertype

of the subtype. As subtypes inherit the characteristics of the instances of their supertype, it is common in conceptual modelling that these inherited characteristics do not need to be specified explicitly for the subtypes.

Third, classification-instantiation relationships are recognized and represented by the relationship between the TYPE and INDIVIDUAL classes in the abstraction pattern (fig. 1). These relationships connect entity types to their instances. The INSTANCE role is assigned to INDIVIDUAL objects and the TYPE role is assigned to TYPE objects (e.g. 'John Doe' (i.e. an individual) is an instance of 'Human' (i.e. a type).).

Fourth, the membership relationship between a SET (i.e. GROUP object) and its MEMBERS (i.e. INDIVIDUAL objects) is shown in the model. This relationship assigns MEMBER roles to INDIVIDUAL objects and SET roles to GROUP objects (e.g. 'John Doe' (i.e. an individual) is a member of the 'Flat Earth Society' (i.e. a group)). This approach to representing sets is also known as **grouping** [1].

Fifth, the abstraction pattern also contains a recursive relationship on the GROUP class that represents subset-superset relationships between GROUP objects, where one GROUP object can be a subset of another GROUP object. For instance, all members of 'Flat Earth Society' (i.e. SUBSET) are also members of 'Society' (i.e. SUPERSET)).

Sixth and finally, Fig. 1 includes a relationship between TYPES and GROUPS. As opposed to the other relationships in this model, it can only be read in one direction (i.e. from type to group). This relationship relates a TYPES object to its extension (i.e. a GROUP object), which is the set of INDIVIDUAL objects that are instances of the TYPE object.

3 Related Work

Part-whole relationships assign PART and WHOLE roles to INDIVIDUAL objects, signifying that in the real world one INDIVIDUAL object is part of another INDIVIDUAL object, which is the whole that the first INDIVIDUAL object is part of. Part-Whole relationships are among the most researched abstraction mechanisms as many authors [5-7] have researched their characteristics (e.g. configurability, separability, mutability), which has led to a classification of a variety of part-whole relationships. UML discriminates two kinds of part-whole relationships (i.e. aggregation and composition). These part-whole relationships can be included in UML class diagrams as an association between two distinct classes (fig. 2a) or a recursive association (fig. 2b). According to our abstraction pattern, part-whole relationships only hold for classes that represent individuals, so not for classes representing types or groups.

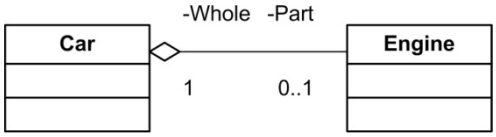

Fig. 2a. Binary Part-Whole relation in UML

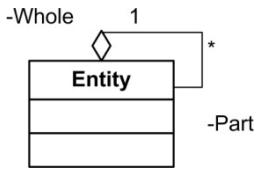

Fig. 2b. Recursive Part-Whole relation in UML

Another abstraction mechanism that is thoroughly addressed in the literature is specialization/generalization. In conceptual modelling, there are two representations of specialization/generalization, each with their own particular semantics. First, using the *power-type* representation [8], subtype-supertype relationships (e.g. an oak is a deciduous tree) are represented by recursive associations on TYPE classes and classification-instantiation relationships (e.g. the tree next to my house is an oak) are represented by binary associations between a TYPE class and an INDIVIDUAL class, called *typification* [1] or *classification* [8] associations (fig. 3a). Second, using a specific UML symbol, i.e. a hollow triangle, subtype-supertype relationships (e.g. a natural person is a specific kind of person) are represented by a direction relationship from the TYPE class representing the subtype to the TYPE class representing the sypertype and classification-instantiation relationships (e.g. 'John Doe' is a natural person) are represented by 'instance-of' dependencies between a TYPE class and an object (fig. 3b). That an instance of a subtype is also an instance of this subtype's supertype, is incorporated in the semantics of UML *generalization* relationships (fig. 3b) whereas it needs to be inferred when using the *power-type* representation (fig. 3a).

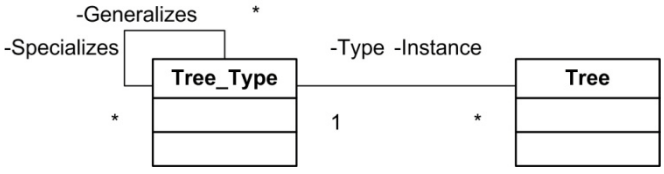

Fig. 3a. Power-Typing

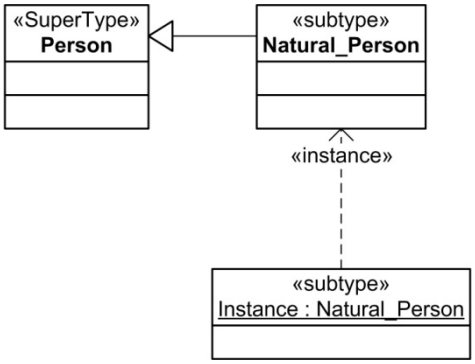

Fig. 3b. UML Generalization relationship

Motschnig-Pitrik and Storey [9] and Brodie [10] studied grouping and the former also addressed the extension relations, identifying different object-level configurations for the abstraction pattern. Fig. 4 shows a possible abstraction pattern instance. More precisely, a group object that is defined by a class (i.e. type) of which the instances are members of the group. There, the instances of M (i.e. M1, M2, M3) are members of G. Therefore, G represents the extension of M.

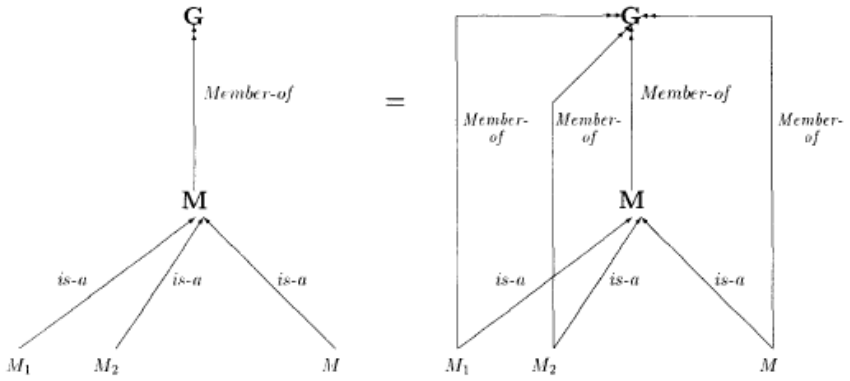

Fig. 4. Object Configuration for Abstraction Pattern (source:[9])

4 Exhibit: PlaneType-Plan-Fleet

In this section, the use of the abstraction pattern for the specification of extension relations between types and groups is demonstrated with the plane-fleet-planeType example by Geerts and McCarthy[1]. Their exhibit (fig. 5a) shows an implicit extension relation between the PLANETYPE and FLEET classes as the FLEET class repeats the 'TYPE NAME' attribute that originates in the PLANETYPE class. That the 'TYPE NAME' attribute does not belong to the FLEET class is clearly shown when the type and grouping characteristics are merged.

Making the extension relation between the PLANETYPE and FLEET classes explicit has the advantage that group definitions can be made explicit. For example, Air France[1] has a fleet that consists entirely of planes (i.e. type). The air france fleet consists of a passenger and a cargo fleet. The passenger fleet is composed of a long range, a middle range and a regional fleet (i.e. all groups extending a certain type of plane (e.g. the long range fleet consists of long range plans)). The longrange fleet consists of 51 Boeing 777, 13 Boeing 747-400, 19 Airbus A340-300 and 15 Airbus A330-200 planes. Each of these groups extends a certain type of longrange plane (e.g. Boeing 777 is-a longrange passenger plane). Next to the Boeing 777, also the the Tupolev Tu-214 is a longrange passenger plane, so although Air France does not have such a plane, it could be considered an alternative for extending the long range fleet.

[1] www.airfrance.com

Due to the consistent mapping of groups and types through the extension relation 'DEFINES', similar hierarchies of groups and types can be constructed (e.g. the Air France fleet consists of planes and the longrange fleet consists of longrange planes of which only a few types of individuals are represented in the fleet).

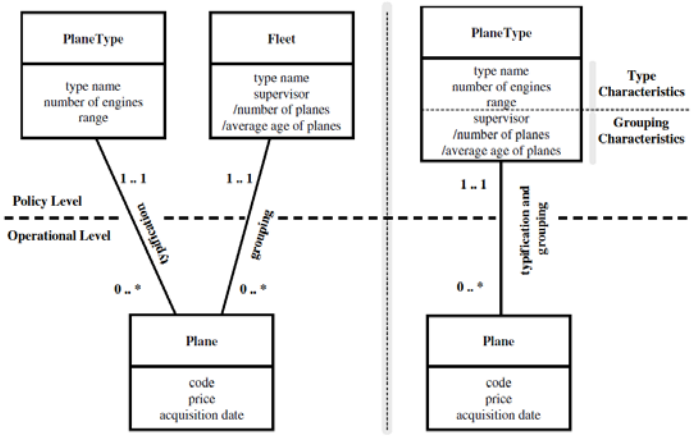

Fig. 5a. Plane-Fleet-PlaneType relations according to Geerts and McCarthy[1]

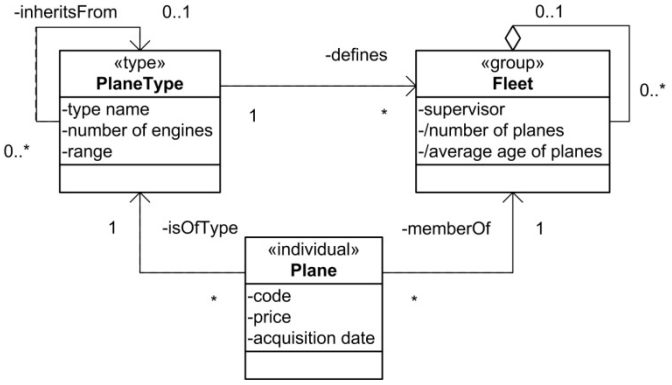

Fig. 5b. Plane-Fleet-PlaneType relations according to abstraction pattern

This simple example shows us that we can represent much more information (e.g. alternative plane types) when we represent extension relationships explicitly.

5 Conclusions & Future Research

In this paper we presented a pattern that addresses the conceptual relationships between different abstraction mechanisms, by positioning them in a foundational ontology. We identified part-whole relationships as conceptual relationships between individuals, and typification and grouping as relationships between individuals and types and groups respectively. Then, supertype-subtype and superset-subset relationships were addressed. The abstraction pattern presented is a particular kind of

root pattern [1], which has its foundations in the UFO ontology and addresses a relation (i.e. extension) that was priorly only addressed implicitly in conceptual modeling literature (e.g. Geerts and McCarthy [1]). For the Plane-Fleet-PlaneType example, we also demonstrated the benefits for the completeness of a model when the abstraction pattern is used as a reference model.

In the future we would like to use this abstraction pattern to provide guidance to modelers, in particular those that develop REA models, and discover and document the forces that drive modellers to use certain abstraction mechanisms and omit relationships when representing policy-level specifications in REA models. These documented forces should then help developing a rule-based modelling method for specifying policies in REA models (e.g. for inexpensive mass-produced resources (e.g. nails) it is common to omit the individual class of the abstraction pattern and merge the type and group attributes in one class [1]). Rule-based approaches appear to outperform pattern-based approaches (like in [1]) for complex modelling tasks in practice [11]. By making these model simplifications explicit and representing them as variations of the abstraction pattern, it should easier to trace and reverse these simplifications if changing requirements (e.g. interoperability with more elaborate data models) requires such a model extension.

References

1. Geerts, G.L., McCarthy, W.E.: Policy-Level Specifications in REA Enterprise Information Systems. Journal of Information Systems **20** (2006) 37-63
2. Borst, W.N.: Construction of engineering ontologies for knowledge sharing an reuse. Centre for Telematics and Information Technology, Vol. doctor. Universiteit Twente, Enschede (1997) 227
3. Guizzardi, G., Wagner, G.: Some Applications of a Unified Foundational Ontology in Business Modeling. In: Green, P., Rosemann, M. (eds.): Business Systems Analysis with Ontologies. Idea group publishing, Hershey PA (2005) xv, 379 p.
4. Chen, P.: The Entity-Relationship model: toward a unified view of data. ACM Transactions on Database Systems **1** (1976) 9-36
5. Barbier, F., Henderson-Sellers, B., Le Parc-Lacayrelle, A., Bruel, J.M.: Formalization of the Whole-Part relationship in the Unified Modeling Language. IEEE Transactions on Software Engineering **29** (2003) 459-470
6. Opdahl, A.L., Henderson-Sellers, B., Barbier, F.: Ontological analysis of whole-part relationships in OO-models. Information and Software Technology **43** (2001) 387-399
7. Guizzardi, G.: Ontological Foundations for Structural Conceptual Models. Vol. Cum Laude. University of Twente (2005)
8. Odell, J.J.: Advanced object-oriented analysis and design using UML. Cambridge University Press ; New York : SIGS, Cambridge (1998)
9. Motschnig-Pitrik, R., Storey, V.C.: Modelling of set membership: The notion and the issues. Vol. 16 (1995) 147-185
10. Brodie, M.L.: Association: A Database Abstraction for Semantic Modelling. Proceedings of the Second International Conference on the Entity-Relationship Approach to Information Modeling and Analysis. North-Holland Publishing Co. (1983)
11. Batra, D., Wishart, N.A.: Comparing a rule-based approach with a pattern-based approach at different levels of complexity of conceptual data modelling tasks. International Journal of Human-Computer Studies **61** (2004) 397-419