AI analysis patterns as UML meta-model constructs

Angeles Manjarrés
Dpto. Int. Artif., UNED,
Senda del Rey s/n, 28040
Madrid, Spain
+33 913988125
amanja@dia.uned.es

Gerson Sunyé, Damien Pollet, Simon Pickin, Jean Marc Jézéquel
IRISA-INRIA,
Campus de Beaulieu,
35042 Rennes, France
+34 (0)2 99 84 74 08
{gsunye, dpollet, spickin, jezequel}@irisa.fr

ABSTRACT

In this article, we investigate the use of the OO computational paradigm for the formulation of knowledge model patterns as OO analysis patterns. We seek to take advantage of research on design pattern specification, aimed at modelling patterns by means of structural and behavioural "meta-level" constraints, introducing appropriate modifications into the UML. We illustrate our argument with the formulation of an OO "assessment pattern" in analogy to the well known "assessment task template".

1. INTRODUCTION

Among the criticisms which could be made of the most commonly-used AI development methodologies, in particular that based on the widely-known KADS-CommonKADS expertise model [2], the suitability of the available techniques for specifying and reusing knowledge-model patterns is one of the most important. The semi-formal notations commonly used do not seem to be adapted to expressing the essence of knowledge-model patterns, and model reuse in the context of these methodologies – based on a task/method/domain trichotomy – is rather cumbersome.

In this article, we investigate the description of AI generic knowledge models ("task templates" in CommonKADS terminology) as OO analysis patterns by applying results obtained in the UML patterns field. In [1][3] a minimal set of modifications to the UML metamodel were proposed to facilitate modelling design patterns and representing their occurrences in UML, opening the way for some automatic processing of pattern applications within CASE tools. The heart of this proposal is to give the UML enough reflexive capacity to allow design patterns to be modelled using structural and behavioural "meta-level" constraints. The role of these constraints in specifying patterns in OO development methodologies is similar to that of the assumptions in specifying generic knowledge models in AI development methodologies. We illustrate this approach by providing a UML description of the assessment pattern, a corresponding KADS-style generic model having been proposed in the literature [2].

2. ASSESSMENT EXPERTISE MODEL

CommonKADS "task templates" consist, firstly, of a task knowledge model and an inference knowledge model characteristic of a problem of a particular type, and secondly, of a specification of a typical domain schema (ontology) that would be required for the application of the corresponding method. In figures 1 and 2 we illustrate how the inference structure of an assessment method (one of the main parts of the task knowledge model) and its correspondent ontology are described for the well-known assessment task template. For more details, including the control structure described in the CML pseudo-code, see [2].

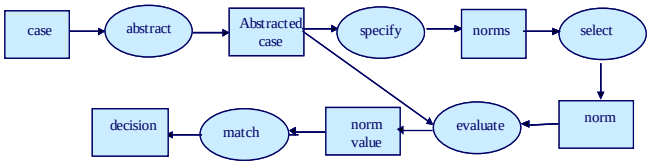

Figure 1. Inference diagram of an assessment method [2]

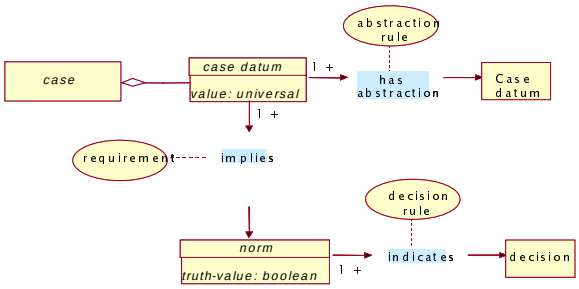

Figure 2. Method-specific ontology [2]

In spite of the fact that these notations are imprecise, they oblige over-specification: the specification of the control structure (as imperative pseudocode) and that of the inferences (informally by their inputs and outputs) may introduce unnecessary constraints on the method and domain, respectively. Moreover, the question of connecting the inference component and the domain component (treated by the so-called assumptions in other approaches based in the expertise model) is not dealt with. The constraints on the binding of the domain concepts to roles exhibit a meta-level character similar to constraints in pattern specification. The task template contains an inference, *evaluate*, to which, in any given application, must be associated a family of inferences with families of associated roles (*norm* and *norm-value*). In the OO design-pattern field, the existence of such families of "class features" can be modelled using the concepts of

Clan and Tribe. Additionally, the descriptions don't include any characterisation of the rule types (see figure 2) used. Rule-types are actually "meta-rules" which cannot be formulated at the expertise-model level since they deal with meta-level concepts. To do so would require languages with reflexive capabilities in the expertise model framework.

3. ASSESSEMENT UML PATTERN

We have discussed some of the problems in representing knowledge-model patterns in the expertise model framework. In the OO design pattern field, similar problems have been identified when trying to represent patterns in a language such as UML [3]. We will apply the pattern-representation approach proposed in [1]. Like UML this approach uses parameterised collaborations to represent patterns, but unlike UML these are represented at the level of UML metamodel. In this way, the role concept can be extended to other model elements, such as methods and attributes (which are classifiers in the metamodel level). This approach is completed by three stereotypes. A «hierarchy» is a set of classes, sharing a common super class. A «clan» is a set of behavioural features that share the same signature and are defined by different classes of the same hierarchy. A «tribe» is set of behavioural features, where each feature is a clan.

For reasons of space, here we concentrate on showing how the ontology and other static aspects of the assessment method are represented. We stress that, in contrast to the task/method/domain trichotomy of the expertise models, an OO pattern is characterised by an unified perspective. The structural aspects of an OO analysis pattern concern both domain knowledge and task & method knowledge. Such a representation, containing classifier roles and association roles, describes the constraints that should be respected by part of a class diagram, i.e. the participants of a pattern occurrence, in order to accomplish the roles defined by the pattern. That is to say, in expertise-model terms, meta-level properties of an "assessment method schema" are described without unnecessarily constraining its structure (e.g. what is called a feature in the UML metamodel, can be an operation, an attribute etc). Certain meta-characteristics of the domain concepts that will assume the specified roles can be expressed using stereotypes.

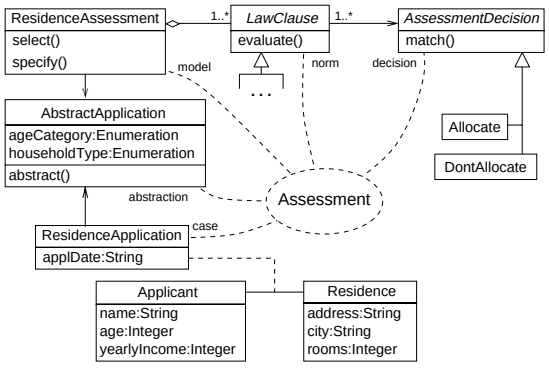

Figure 3. Class model in "the housing application"

Figure 3 shows a UML representation of an occurrence of the assessment pattern; based on the instantiation of the assessment task template specified in [2]. The collaboration usage is symbolised by an ellipse. The links between the collaboration usage and the classes represent the roles played by each class.

The main part of the structural representation of the assessment pattern is shown in Figure 4. The main participant class is represented by the *Model* classifier role, which owns two role features, *select* and *specify*. This means that, in the occurrence example of figure 3, the class Residence Assessment must implement at least two methods (corresponding to these roles). Since these are roles and not real methods, different names could be used.

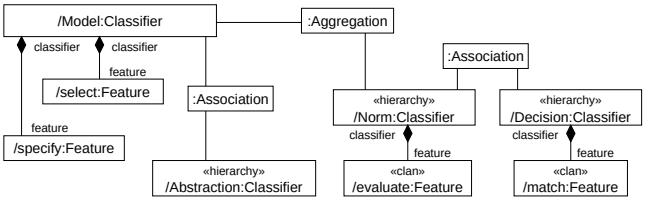

Figure 4. Meta-level Collaboration for the assessment pattern.

The role *Model* has two relationships, with two different classifier roles, *Abstraction* and *Norm*, both stereotyped «hierarchy», which means that theses roles can be played by a hierarchy of classes. Indeed, in the example above, several clauses can be used to determine if a residence can be allocated to an applicant. Since each clause may overload the *evaluate*() method, the *evaluate* role is stereotyped «clan». The role *Norm* has an association with *Decision*, which is also a hierarchy.

The representation of the Assessment pattern is completed by a set of additional OCL meta-level constraints characterising the rule types and the inference knowledge of the assessment template) These either constrain elements of this extended metamodel or, more commonly, any instantiation of it. These "static" constraints could be completed with some temporal constraints specifying dynamic aspects of the assessment method (for instance, that there exists a precise order to the features to be called, that is, abstract, specify, select and match). It is important to note that this last constraint cannot currently be expressed in OCL. We are currently investigating possible solutions as the use of some form of temporal logic or the adaptation of UML sequence diagrams, so that they could describe behavioural constraints at the required level of abstraction.

4. CONCLUSIONS

Objections to the most commonly-used AI analysis methodologies have led us to the formulation of knowledge model patterns as OO analysis patterns. We have taken advantage of valuable research work done in the design pattern specification field. A library of these OO patterns (a "pattern language") could constitute the counterpart of the widespread AI analysis libraries.

5. REFERENCES

[1] Le Guennec, A., Sunyé, G. Jézéquel, J.M., Precise Modeling of Design Patterns, in Proceedings of UML 2000, LNCS 1939, Springer Verlag, 2000.

[2] Schreiber, G., Akkermans, H., Anjewierden, A., de Hoog, R., Shadbolt, N., Van de Velde, W., Wielinga, B.J.. Knowledge Engineering and Management. The CommonKADS Methodology. MIT Press, 2000.

[3] Sunyé, G. Le Guennec A., Jézéquel, J.M.. Design Pattern Application in UML, in Proceedings of ECOOP'2000, LNCS 1850, Springer Verlag, 2000.