Usability of a Visual Language for DL Concept Descriptions*

Fernando Náufel do Amaral

LLaRC – Laboratório de Lógica e Representação do Conhecimento,
Depto. de Ciência e Tecnologia, Pólo Universitário de Rio das Ostras,
Universidade Federal Fluminense, Rio das Ostras, RJ, Brazil
fnaufel@ic.uff.br

Abstract. The development and use of ontologies may require users with no training in formal logic to handle complex concept descriptions. To aid such users, we propose a new visualization framework called "model outlines", where more emphasis is placed on the *semantics* of concept descriptions than on their *syntax*. We have conducted a usability study comparing model outlines and Manchester OWL, with results that indicate the potential benefits of our visual language for understanding concept descriptions.

1 Introduction

When working with formal ontologies, one often needs to formally represent conditions for membership in the defined classes. In this paper, we will call such conditions *concept descriptions*, following the description logic (DL) tradition [1].

Concept descriptions are important in many scenarios related to ontology development and use. For example, DL reasoners perform logical inferences by manipulating concept descriptions according to a specific deductive calculus. In many cases, users may be interested not only in the answers provided by such reasoners, but also in the chains of reasoning that led to those answers. In order to understand such chains of reasoning, users must be able to understand the meaning of the concept descriptions involved. This area of study is referred to as *proof explanation* [2].

Another situation where concept descriptions play an important role is in the definition and use of *ontology query languages* [3]; here, building a query may include writing modified concept descriptions that contain free variables (representing individuals that must be returned by the query).

Because many users of formal ontologies have no specific training in logic, the problem of representing concept descriptions in a user-friendly fashion is an important one, and many researchers have proposed different ways of solving it: replacing logical symbols with keywords in DL languages [4], automatically generating natural language paraphrases of concept descriptions [5], or using diagrams [6, 7].

* Work partially supported by research grant E-26/112.038/2008 from FAPERJ.

As an example to make this discussion more concrete, consider the following concept description in DL syntax (to be formally introduced in Sect. 2 below), which appears in [8], a paper about proof explanation:

$$\exists hasChild.\top \sqcap \forall hasChild.\neg((\exists hasChild.\neg Doctor) \sqcup (\exists hasChild.Lawyer)) \quad (1)$$

Diagrammatic representations of concept descriptions have given rise to implementations of "visual" ontology browsers. One such example is the visualization tool GrOWL [9], which produces the diagram in Fig. 1 for the concept description in (1). As can be seen, the diagram is essentially an abstract syntax tree, which offers nonspecialist users little help in understanding the semantics of the description, especially if those users are not familiar with the DL symbols "$\exists$", "$\forall$", "$\neg$" and "$\sqcup$". In fact, we have found this to be a common phenomenon: many visualization frameworks for concept descriptions are too faithful to the *syntax* of the representation languages (e.g., DL, OWL), a feature which may prevent users from grasping the *semantics* of the concept descriptions.

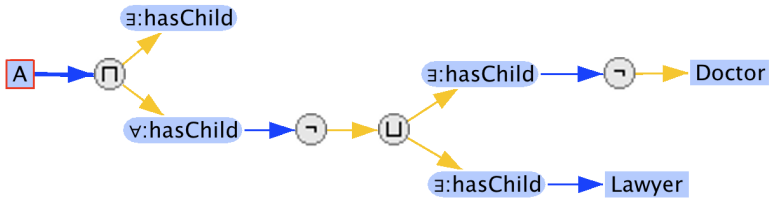

Fig. 1. Diagram produced for (1) by GrOWL [9] (manually laid out)

This paper discusses *model outlines*, which depart from the syntax-based tradition in that they consist of diagrams characterizing the class of *models* of a given concept description. (Here, we use the term "model" in the logical sense.) The model outline for (1), produced after applying a carefully defined set of simplification rules to the original concept description, is presented in Fig. 2. By adhering to some simple graphical conventions, a user can understand that the concept description represents a set of individuals having at least one child and having as grandchildren (if any) only doctors and non-lawyers.

Our previous papers [10, 11] introduced the first version of model outlines and compared them to natural language paraphrases of concept descriptions. Since then, we have reformulated the visual language so as to make it more intuitive (e.g., including optional labeled clusters, rendering cardinality restrictions as text and fine-tuning the placement of inner boxes). We have also altered the conversion algorithms to conform to the new visual language.

Most importantly, we have conducted a first usability test of model outlines, with promising results. Users from different backgrounds were shown concept

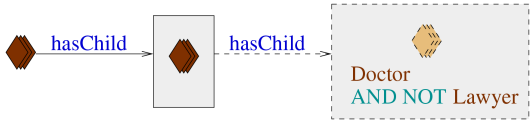

Fig. 2. Model outline for description (1)

descriptions in two formalisms: our model outlines and Manchester OWL (a textual notation for DL which uses keywords for logical symbols, infix notation for restrictions, syntax highlighting and indentation in order to make descriptions more readable for nonspecialists — see [4]). We then tested ease of understanding for each formalism by asking the users questions about the concept descriptions shown.

This paper is structured as follows: Sect. 2 presents the syntax of model outlines for the description logic $\mathcal{ALCN}$, at the concrete (token) and at the abstract (type) levels, as is suitable for diagrammatic systems [12]; Sect. 3 defines the precise semantics of model outlines, in the form of algorithms that translate from model outlines to $\mathcal{ALCN}$ concept descriptions; Sect. 4 discusses the translation of $\mathcal{ALCN}$ concept descriptions to model outlines; Sect. 5 reports and analyzes the results of the usability test; Sect. 6 contains our concluding remarks.

2 Syntax of model outlines

We consider the description logic $\mathcal{ALCN}$, whose language of concept descriptions[1] is specified in Fig. 3, both in the DL syntax and in Manchester OWL. There, A stands for a class name (i.e., an atomic concept term), R stands for a property name (i.e., an atomic role term), and n represents a natural number. The (set-theoretical) meaning of these descriptions is given by a nonempty set Δ (the *universe* or *domain*) along with an interpretation $\mathcal{I}$ mapping each concept description C to a set $\mathcal{I}(C) \subseteq \Delta$, and each role term R to a binary relation $\mathcal{I}(R) \subseteq \Delta \times \Delta$. An interpretation $\mathcal{I}$ must map each description in the first two columns to the set in the third column. $\#S$ denotes the cardinality of a set S. A *literal* is a description of the form A or of the form $\neg A$, where A is an atomic concept term.

The *concrete syntax* of model outlines defines their physical representation. What follows is an informal definition: a model outline contains *clusters* (*solid* or *dashed*), *arrows* (*solid* or *dashed*) and *boxes*. The *root* of the model outline is a solid cluster. A cluster may have an optional *class label* below it, consisting of a disjunction or of a conjunction of literals. So may a box. A box may also have an optional *cardinality label* below it, which may be of the form "(from m thru n)", "(m or more)", or "(exactly m)", with m, n natural numbers, $m < n$. The *source* of

[1] Work is under way to define model outlines for more expressive languages, such as the concept language underlying OWL 2 [13].

DL	Manchester	Meaning
$C, D \rightarrow A$	A	$\mathcal{I}(A)$
$\mid \top$	**THING**	Δ
$\mid \bot$	**NOTHING**	$\varnothing$
$\mid \neg C$	**NOT** C	$\Delta - \mathcal{I}(C)$
$\mid C \sqcap D$	C **AND** D	$\mathcal{I}(C) \cap \mathcal{I}(D)$
$\mid C \sqcup D$	C **OR** D	$\mathcal{I}(C) \cup \mathcal{I}(D)$
$\mid \forall R.C$	R **ONLY** C	$\{a \in \Delta \mid \forall b.[(a,b) \in \mathcal{I}(R) \Rightarrow b \in \mathcal{I}(C)]\}$
$\mid \exists R.C$	R **SOME** C	$\{a \in \Delta \mid \exists b.[(a,b) \in \mathcal{I}(R) \wedge b \in \mathcal{I}(C)]\}$
$\mid \leq n.R$	R **MAX** n	$\{a \in \Delta \mid \#\{b \mid (a,b) \in \mathcal{I}(R)\} \leq n\}$
$\mid \geq n.R$	R **MIN** n	$\{a \in \Delta \mid \#\{b \mid (a,b) \in \mathcal{I}(R)\} \geq n\}$
$\mid = n.R$	R **EXACTLY** n	$\{a \in \Delta \mid \#\{b \mid (a,b) \in \mathcal{I}(R)\} = n\}$

Fig. 3. $\mathcal{ALCN}$ concept descriptions and their meaning

an arrow may be a cluster or a box. The *target* of an arrow is always a box. Each box is the target of exactly one arrow. An arrow must have a *role label* above it, consisting of a role name. A box *contains* one or more clusters, according to constraints that we do not include in this informal description, but which will be made explicit in the abstract syntax below. A box may also contain at most one *"among-which" inner box*, which in turn contains one or more clusters, all of them solid. Inner boxes are never the source of arrows. A box or a cluster may have a *case widget* above it.

Fig. 4 shows an example model outline. The target box of the arrow labeled "*hasAttendance*" has both a class label ("*Enrolled*") and a cardinality label ("from 10 to 50"). The target box of the arrow labeled "*hasAttendance*" also has an "among-which" inner box. This model outline does not have case widgets.

At this point, the reader should test the appropriateness of the choice of visual presentation of the components of model outlines. We suggest that the reader (without any further knowledge of the meaning of these components) formulate a natural language description of the constraints imposed upon the individuals of class *GraduateCourse* at the root of the outline. If the reader is knowledgeable in DL syntax, the reader should also produce an $\mathcal{ALCN}$ concept description. In Sect. 3 below, we explain the precise meaning of this model outline, and in Sect. 4 we show the steps involved in its construction.

Case widgets indicate alternatives (i.e., disjunction). If a cluster or a box has a case widget above it, the user may browse the different cases interactively, one case at a time, by clicking on the triangles on either side of the case widget.

In Fig. 5, for example, there are 4 cases altogether, specifying objects that are either (a) *Book*s having all extras (if any) translated to *Portuguese* (and possibly other languages), or (b) *Book*s having all extras (if any) in *Audio* format (and possibly other formats), or (c) *ClassNotes* having at least one *Free* copy in *PDF* format (and possibly other formats, and other copies), or (d) *ClassNotes* having at least one *Low*-priced copy (and possibly other copies).

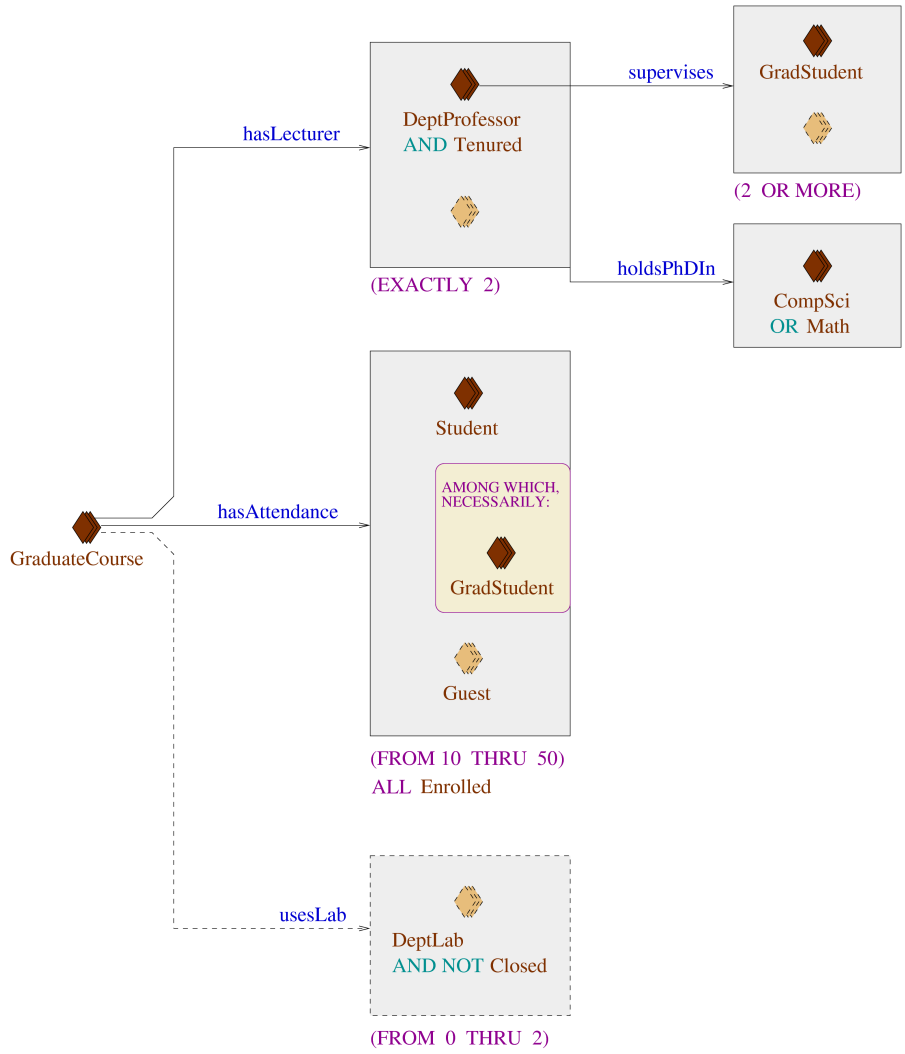

Fig. 4. Example model outline

More formally, the model outline in Fig. 5 corresponds to the description

$$[Book \sqcap \forall hasExtras.(\exists hasTranslation.Portuguese \sqcup \exists hasFormat.Audio)] \sqcup$$
$$\{ClassNotes \sqcap$$
$$\exists hasCopy.[(Free \sqcap \exists hasFormat.PDF) \sqcup (\exists hasPrice.Low \sqcap \forall hasPrice.Low)]\}$$

As for the *abstract syntax*, a model outline is formally defined as a LISP-style list generated by the grammar in Fig. 6, in extended BNF notation. The

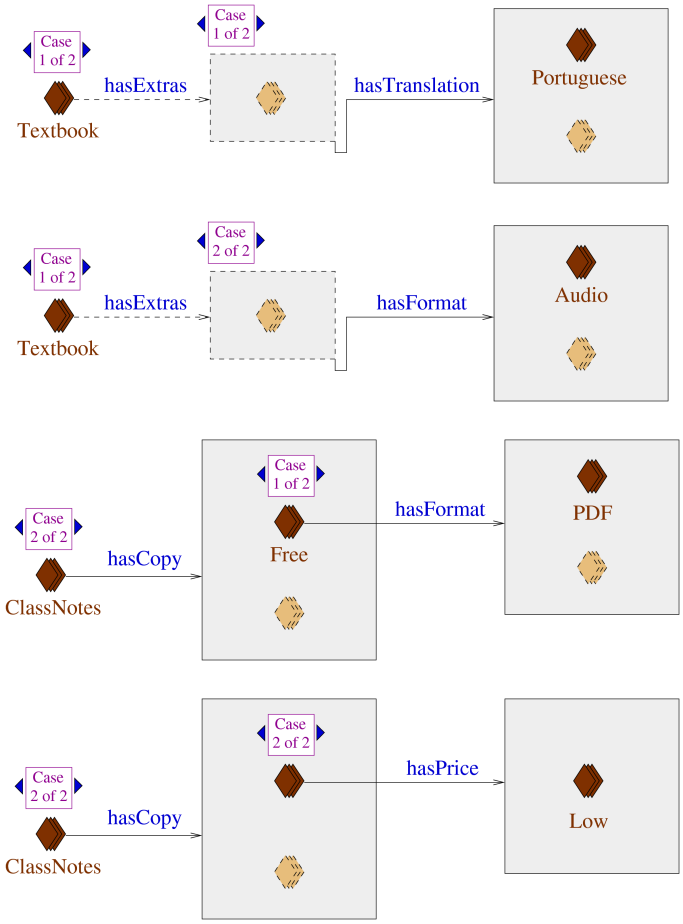

Fig. 5. Example model outline with case widgets

list representation is not meant for human consumption, but rather for automatic processing by algorithms such as the ones presented in the next section.

3 Semantics of model outlines

The appearance of the components of a model outline follows some (hopefully intuitive) graphical conventions:

Individuals are represented by clusters of diamonds. The presence of a cluster (as opposed to a single diamond) emphasizes the idea that one *or more* individuals may appear in a given situation. E.g., in Fig. 4, the graduate courses in question may have as lecturers more than one tenured department professor

$$
\begin{aligned}
\langle outline\rangle &\rightarrow \langle solidClstrCases\rangle\\
\langle solidClstrCases\rangle &\rightarrow \textbf{(}\ \textbf{cases}\ \langle solidCluster\rangle^{+}\ \textbf{)}\\
\langle solidCluster\rangle &\rightarrow \textbf{(}\ \textbf{cluster solid}\ \langle classLabel\rangle\ \textbf{(}\ \langle arrow\rangle^{\star}\ \textbf{)}\ \textbf{)}\\
\langle classLabel\rangle &\rightarrow \textbf{(}\ \textbf{)} \mid \textbf{(}\ \langle literal\rangle\ \textbf{)}\\
&\quad \mid \textbf{(}\ \textbf{and}\ \langle literal\rangle\ \langle literal\rangle^{+}\ \textbf{)} \mid \textbf{(}\ \textbf{or}\ \langle literal\rangle\ \langle literal\rangle^{+}\ \textbf{)}\\
\langle literal\rangle &\rightarrow \langle conceptName\rangle \mid \textbf{(}\ \textbf{not}\ \langle conceptName\rangle\ \textbf{)}\\
\langle arrow\rangle &\rightarrow \textbf{(}\ \textbf{arrow solid}\ \langle roleName\rangle\ \textbf{(}\ \langle intrvl\rangle^{\star}\ \textbf{)}\ \langle solidBoxCases\rangle\ \textbf{)}\\
&\quad \mid \textbf{(}\ \textbf{arrow dashed}\ \langle roleName\rangle\ \textbf{(}\ \langle intrvl\rangle^{\star}\ \textbf{)}\ \langle dashedBoxCases\rangle\ \textbf{)}\\
\langle intrvl\rangle &\rightarrow \textbf{(}\ \langle number\rangle\ \langle number\rangle\ \textbf{)} \mid \textbf{(}\ \langle number\rangle\ \textbf{infty}\ \textbf{)}\\
\langle solidBoxCases\rangle &\rightarrow \textbf{(}\ \textbf{cases}\ \langle solidBox\rangle^{+}\ \textbf{)}\\
\langle solidBox\rangle &\rightarrow \textbf{(}\ \textbf{box}\ \langle classLabel\rangle\ \textbf{(}\ \langle solidClstrCases\rangle^{+}\ \textbf{)}\ \langle opt\rangle\ \textbf{(}\ \langle arrow\rangle^{\star}\ \textbf{)}\ \textbf{)}\\
\langle opt\rangle &\rightarrow \textbf{(}\ \langle unlabeledCluster\rangle\ \textbf{)} \mid \textbf{(}\ \langle innerBox\rangle?\ \langle dashedClstrCases\rangle?\ \textbf{)}\\
\langle unlabeledCluster\rangle &\rightarrow \textbf{(}\ \textbf{cluster dashed (}\ \textbf{)}\ \textbf{(}\ \textbf{)}\ \textbf{)}\\
\langle innerBox\rangle &\rightarrow \textbf{(}\ \textbf{innerBox}\ \langle solidClstrCases\rangle^{+}\ \textbf{)}\\
\langle dashedBoxCases\rangle &\rightarrow \textbf{(}\ \textbf{cases}\ \langle dashedBox\rangle^{+}\ \textbf{)}\\
\langle dashedBox\rangle &\rightarrow \textbf{(}\ \textbf{box}\ \langle classLabel\rangle\ \textbf{(}\ \langle snglDashedCluster\rangle\ \textbf{)}\ \textbf{(}\ \textbf{)}\ \textbf{(}\ \langle arrow\rangle^{\star}\ \textbf{)}\ \textbf{)}\\
\langle snglDashedCluster\rangle &\rightarrow \textbf{(}\ \textbf{cases (}\ \textbf{cluster dashed}\ \langle classLabel\rangle\ \textbf{(}\ \textbf{)}\ \textbf{)}\ \textbf{)}\\
\langle dashedClstrCases\rangle &\rightarrow \textbf{(}\ \textbf{cases}\ \langle dashedCluster\rangle^{+}\ \textbf{)}\\
\langle dashedCluster\rangle &\rightarrow \textbf{(}\ \textbf{cluster dashed}\ \langle classLabel\rangle\ \textbf{(}\ \langle arrow\rangle^{\star}\ \textbf{)}\ \textbf{)}
\end{aligned}
$$

Fig. 6. Abstract, formal syntax for $\mathcal{ALCN}$ model outlines

holding a CompSci or Math PhD degree and supervising at least one graduate student from a total of 2 or more individuals.

Clusters of *solid* diamonds represent individuals that must exist. In Fig. 4, it is mandatory that the graduate courses in question have as lecturer at least one tenured department professor holding a CompSci or Math PhD degree and supervising at least one graduate student from a total of 2 or more individuals. Likewise, the attendance must include students and graduate students.

Clusters of *dashed* diamonds represent optional individuals. If the cluster is labeled or has outgoing arrows, the individuals must belong to the corresponding class (e.g., "*Guest*" in Fig. 4). If the cluster is unlabeled, the individuals may belong to any class, subject to the constraints stipulated by the label and the outgoing arrows of the outer box where the cluster is located (e.g., in Fig. 4, the unlabeled cluster in the "*hasLecturer*" box represents lecturers that do not have to be tenured department professors, but that must hold a CompSci or Math PhD degree).

As indicated in the previous remark, *box labels and arrows originating from boxes* represent constraints that must be satisfied by all individuals corresponding to clusters in the box. In Fig. 4, all individuals attending the graduate courses in question must belong to class "*Enrolled*".

The *absence of a dashed cluster* in a box means that all the individuals represented in the box must belong to the classes specified by their respective labels and to the class specified by the box label and arrows (if present). This

is evident in Fig. 4, where it is required that the lecturers hold a PhD degree *only* in CompSci or Math (a rather exclusivist and unfair requirement, but this is only an example).

Dashed boxes, always the target of dashed arrows, always contain a dashed cluster, representing optional individuals. In Fig. 4, the graduate courses in question may or may not involve the use of (up to 2) department labs.

"Among which" inner boxes contain clusters representing individuals that belong to subclasses of one or more classes specified in the outer box. In Fig. 4, the attendance of the graduate courses in question consists of students, some of which are required to be graduate students. Optionally, guests may attend.

The above remarks are included here only for pedagogical purposes. In fact, we define the precise semantics of model outlines by means of algorithm DESCR, which, when given a model outline C (in abstract syntax), yields the $\mathcal{ALCN}$ concept description taken as the meaning of C. Algorithm DESCR calls BOXDESCR to build the concept description denoted by a box. Fig. 7 shows both algorithms.

The reader should refer to the grammar in Fig. 6 for the structure of the lists that the algorithms manipulate. These algorithms can be modified to produce more legible output; here, their only purpose is to serve as the precise semantics of model outlines. When given as input the model outline in Fig. 2, e.g., algorithm DESCR returns the following description, which is equivalent to (1):

$$\bot \sqcup \{\top \sqcap \forall \mathit{hasChild}.(\bot \sqcup \bot \sqcup \top) \sqcap \top \sqcap \exists \mathit{hasChild}.(\bot \sqcup \top) \\ \sqcap \forall \mathit{hasChild}.[\top \sqcap (\bot \sqcup \forall \mathit{hasChild}.(\bot \sqcup \bot \sqcup (\mathit{Doctor} \sqcap \neg \mathit{Lawyer})))]\}$$

4 Constructing model outlines

We have presented elsewhere [10] detailed algorithms for translating $\mathcal{ALCN}$ concept descriptions into model outlines. Here, we incorporate some changes to the algorithms (e.g., to account for labeled optional clusters) and give a more informal explanation of the main steps involved in such a translation, using as a working example the concept description that originated the model outline in Fig. 4.

Given an $\mathcal{ALCN}$ concept description C, we start by converting C to modified disjunctive normal form (mDNF), applying simplification rules in the process. A concept description is in mDNF if it fits the pattern

$$D_1 \sqcup \ldots \sqcup D_n$$

where each disjunct D_i is a conjunction of the form

$$C_1 \sqcap \cdots \sqcap C_p$$

where each conjunct C_j is either a literal, or a collection of "intervals" of natural numbers (whose upper bound may be ∞) associated to a role R, or a description of the form $\forall R.C'$ or of the form $\exists R.C'$, where C' is itself in mDNF.

```
DESCR(C)                                   ▷ C has the form ( cases C_1 ··· C_m )
1  Descr ← ⊥
2  for each C_i in C_1, ..., C_m           ▷ C_i has the form ( cluster S L ( A_1 ··· A_n ) )
3       do if L = ( )
4             then Case ← ⊤
5             else  Case ← L
6          for each A_j in A_1, ..., A_n
7               do Case ← Case ⊓ BOXDESCR(A_j)
8          Descr ← Descr ⊔ Case
9  return Descr

BOXDESCR(A)    ▷ A has the form ( arrow S RN ( I_1 ··· I_n ) ( cases B_1 ··· B_m ) )
 1  BDescr ← ⊥
 2  if n = 0                                              ▷ No cardinality restrictions
 3     then Card ← ⊤
 4     else  Card ← ⊥
 5           for each I_j in I_1, ..., I_n               ▷ I_j is "interval" of the form ( X Y )
 6                do if Y = infty
 7                      then Card ← (Card ⊔ ≥ X.RN)
 8                      else  Card ← (Card ⊔ (≥ X.RN ⊓ ≤ Y.RN))
 9  for each B_i in B_1 ··· B_m
10     ▷ Each box case B_i has the form ( box BL ( C_1 ··· C_p ) Opt ( A'_1 ··· A'_q ) )
11        do Universal ← ⊥; Existentials ← ⊤
12           for each C_j in C_1, ..., C_p                                  ▷ Cluster cases
13                do Universal ← Universal ⊔ DESCR(C_j)
14                   if C_j has the form ( cluster solid ... )
15                      then Existentials ← Existentials ⊓ ∃RN .DESCR(C_j)
16           if Opt contains ( innerBox C'_1 ··· C'_r )
17              then for each C'_j in C'_1, ..., C'_r
18                        do Existentials ← Existentials ⊓ ∃RN .DESCR(C'_j)
19           if Opt contains ( cluster dashed ( ) ( ) )
20              then Universal ← ⊤
21           if Opt contains ( cases C''_1 ··· C''_s )                      ▷ Optional clusters
22              then for each C''_j in C''_1, ..., C''_s
23                        do Universal ← Universal ⊔ DESCR(C''_j)
24           Universal ← ∀RN.(Universal)
25           if BL = ( )
26              then BCase ← Universal ⊓ Existentials
27              else  BCase ← ∀RN.BL ⊓ Universal ⊓ Existentials
28           for each A'_j in A'_1, ..., A'_q                               ▷ Box arrows
29                do BCase ← BCase ⊓ ∀RN.BOXDESCR(A'_j)
30           BDescr ← BDescr ⊔ BCase
31  return Card ⊓ BDescr
```

Fig. 7. Algorithms to convert from model outlines to $\mathcal{ALCN}$

The modification is in the way number restrictions are represented: using appropriate rewrite rules, any conjunction of cardinality restrictions over a role R_i can be converted to a collection of "intervals" of natural numbers.[2]

To each D_i we then apply the simplification rule

$$\forall R.C_1 \sqcap \ldots \sqcap \forall R.C_n \ \rhd \ \forall R.(C_1 \sqcap \ldots \sqcap C_n)$$

As a result, we obtain C', which is a disjunction $D'_1 \sqcup \ldots \sqcup D'_n$, where each D'_i can be written as

$$L_1 \sqcap \ldots \sqcap L_m \sqcap C_1 \sqcap \cdots \sqcap C_p$$

where each L_i is a literal, and each C_j can be written as

$$\forall R.F \sqcap \exists R.G_1 \sqcap \ldots \sqcap \exists R.G_q \sqcap K$$

where F and all the G_i are in mDNF and K is a collection of intervals of natural numbers representing cardinality restrictions over role R. Any (or all) of these elements may be absent. Note that we have grouped the conjuncts according to the role R they refer to. Later, when the model outline is built, each of these groups will originate an arrow labeled by R.

Following these guidelines, the simplified mDNF of the concept description corresponding to the example model outline in Fig. 4 is found to be

$$\begin{aligned}
&\mathit{GraduateCourse} && \text{(2a)}\\
\sqcap\ &\forall \mathit{hasLecturer}. && \text{(2b)}\\
&\quad [\forall \mathit{holdsPhDIn}.(\mathit{CompSci} \sqcup \mathit{Math}) && \text{(2c)}\\
&\quad\ \sqcap \exists \mathit{holdsPhDIn}.(\mathit{CompSci} \sqcup \mathit{Math})] && \text{(2d)}\\
\sqcap\ &\exists \mathit{hasLecturer}.(\mathit{DeptProfessor} \sqcap \mathit{Tenured} \sqcap \exists \mathit{supervises}.\mathit{GradStudent} && \text{(2e)}\\
&\qquad\qquad \sqcap \{[2,\infty]\}.\mathit{supervises}) && \text{(2f)}\\
\sqcap\ &\{[2,2]\}.\mathit{hasLecturer} && \text{(2g)}\\
\sqcap\ &\forall \mathit{hasAttendance}.[(\mathit{Student} \sqcap \mathit{Enrolled}) \sqcup (\mathit{Guest} \sqcap \mathit{Enrolled})] && \text{(2h)}\\
\sqcap\ &\exists \mathit{hasAttendance}.\mathit{Student} && \text{(2i)}\\
\sqcap\ &\exists \mathit{hasAttendance}.\mathit{GradStudent} && \text{(2j)}\\
\sqcap\ &\{[10,50]\}.\mathit{hasAttendance} && \text{(2k)}\\
\sqcap\ &\forall \mathit{usesLab}.(\mathit{DeptLab} \sqcap \neg \mathit{Closed}) && \text{(2l)}\\
\sqcap\ &\{[0,2]\}.\mathit{usesLab} && \text{(2m)}
\end{aligned}$$

Note how the constraints have been grouped by the roles they act upon. Note also how the cardinality constraints in lines (2f), (2g), (2k) and (2m) have been written with (singleton) collections of intervals of natural numbers.

Two transformations must be effected before the model outline can be built.

[2] For role R, the interval $[m,n]$ represents the constraint $(\geq m.R \sqcap \leq n.R)$. Likewise, $[m,m]$ represents $(= m.R)$, and $[0,m]$ represents $(\leq m.R)$, and $[m,\infty]$ represents $(\geq m.R)$.

The first one concerns lines (2b)–(2d), where the set of objects related to the lecturers through *holdsPhDIn* is *closed*: i.e., the lecturers must hold *some* PhD degree in CompSci or Math and *only* PhD degrees in CompSci or Math.

The algorithm detects such a closure whenever it finds conjuncts of the form

$$\forall R(C_1 \sqcup \cdots \sqcup C_n) \sqcap \exists R.C_1 \sqcap \cdots \sqcap \exists R.C_n$$

Here, we have $n = 1$ and $C_1 = \mathit{CompSci} \sqcup \mathit{Math}$. Then, to indicate the closure, the algorithm refrains from adding a dashed, unlabeled cluster to the target box of the *holdsPhDIn* arrow (see Fig. 4).

The second transformation is similar: in lines (2h)–(2j), we can see there is some sort of closure related to the role *hasAttendance*, but the situation is more complicated. In fact, this is the general case, which also includes the first transformation. Whenever the conjuncts for role R are of the form

$$\begin{aligned}&\forall R[(C_1 \sqcap D) \sqcup \cdots \sqcup (C_n \sqcap D) \sqcup (C_{n+1} \sqcap D) \sqcup \cdots \sqcup (C_{n+p} \sqcap D)] \\ &\sqcap \exists R.C_1 \sqcap \cdots \sqcap \exists R.C_n \sqcap \exists R.F_1 \sqcap \cdots \sqcap \exists R.F_q\end{aligned}$$

where D is a conjunction (with $D = \top$ as the trivial case) it proceeds as follows:

- Solid clusters for $C_1, \ldots, C_n$ are created in the main target box for the R-arrow.
- The main target box for the R-arrow gets D as a label. If $D = \top$, this label is not shown.
- The main target box for the R-arrow gets an "among which" inner box containing solid clusters for $F_1, \ldots, F_q$.
- Dashed clusters for $C_{n+1}, \ldots, C_p$ are created in the main target box for the R-arrow.

In our example description, in lines (2h)–(2j), we have that $n = 1$, and $C_1 = \mathit{Student}$, and $D = \mathit{Enrolled}$, and $p = 1$, and $C_2 = \mathit{Guest}$, and $q = 1$, and $F_1 = \mathit{GradStudent}$.

5 Evaluation

We have conducted a usability study in order to evaluate our proposed diagrammatic notation. The main aim was to *test the usefulness of model outlines for the understanding of complex concept descriptions.*

Note that it is the model outline *notation* itself that is being evaluated, not a specific graphical user interface (GUI) implementing the notation. Thus, the focus of the study is on understanding, not on interaction. We find this to be an advantage, as changes can be made to the notation before we are committed to a specific GUI, and problems can be identified in relation to specific features of the notation, so that special attention can be given to these problems in order to solve or mitigate them through the use of appropriate human interaction techniques. From a practical point of view, this potentially reduces the need for radical, costly changes after implementation.

Likewise, we have chosen model outlines for the simpler $\mathcal{ALCN}$ language so we could find out early if something needs to be changed in our most basic assumptions. The result of this test will help us design the extensions of model outlines to deal with more expressive concept languages.

Following [14], we defined our *main goal* as: *Model outlines can help users with little or no training in Logic to understand complex concept descriptions. In particular, model outlines are more effective than Manchester OWL for this task.*

Manchester OWL (see Fig. 3 and also [4]) is a textual notation for DL which uses keywords for logical symbols (e.g., "**SOME**" for "$\exists$"), infix notation for restrictions (e.g., "*hasChild* **SOME** *Man*" for "$\exists hasChild.Man$"), syntax highlighting and indentation in order to make descriptions more readable for non-specialists. So, we are comparing our diagrammatic notation with a textual notation designed for the same target audience. (As the test participants were all Brazilians, we used Portuguese translations of the Manchester OWL keywords.)

Next, we defined a set of *concerns*, in the form of questions like: *Can users understand the meaning of X?*, where X is one of the elements present in model outlines (solid clusters, dashed clusters, arrows, boxes, inner boxes, case widgets, etc.). Specific concerns were also formulated (e.g., "Can users understand that individuals in "among which" inner boxes are mandatory?").

We selected 10 participants for our study ([14] recommends 6 to 12). These participants come from several backgrounds and occupations, as detailed below. All received detailed information on the procedures and on their rights as participants. All signed terms of informed consent.

One session of the study consisted of the following activities: a pre-test questionnaire, a tutorial on notation A, a specification on domain X using notation A, 15 questions, a post-task questionnaire, a tutorial on notation B, a specification on domain Y using notation B, 15 questions, a post-task questionnaire, and a post-test questionnaire. Notations A and B alternated between model outlines and Manchester OWL. Domains X and Y alternated between graduate courses (which included Fig. 4 of this paper) and family relations. Each participant answered 15 questions for each domain. The questions for each domain were fixed, regardless of the notation used. For each domain, half the participants answered questions on model outlines, and half answered questions on Manchester OWL specifications. Half the participants saw model outlines before Manchester OWL, and half saw Manchester OWL before model outlines.

The number of correct answers and the time to answer were measured. Additional information was obtained in the form of comments collected through the "thinking out loud" protocol [14] and through questionnaires. Table 1 shows the occupation and the number of correct answers for each participant:

For the graduate courses domain, we note the following highlights:

Question 8 was related to Fig. 4 of this paper, and elicited *5 errors* using Manchester OWL, and *no errors* using model outlines. The question was: "If a course is attended only by students that are *not* graduate students, does the course meet the specification?" The error was probably induced by the abbrevia-

Occupation	Correct answers (model outlines)	Correct answers (Manchester OWL)
Logician	15	14
Theoretical physicist	15	12
Software engineer	15	12
Secretary	15	10
Nurse	13	12
Graphics designer	13	12
Social worker	13	11
Comp. Science undergrad	13	10
Production engineer	13	9
Mathematician	12	14
Totals:	137	116
Percentages:	91.3%	77.3%

Table 1. Occupation and number of correct answers for each participant

tion recommended in [4]: "*hasAttendance* **SOME** [*Student*, *GradStudent*]", which seems to have evoked the idea that the bracketed list consisted of a set of alternatives. This question was answered correctly by all participants using model outlines, which indicates that users understood the meaning of "among which" inner boxes.

Question 14 elicited *4 errors* using Manchester OWL, and *3 errors* using model outlines. This question was about a specification consisting of 4 cases. The situation proposed in the question satisfied exactly one of the 4 cases. With Manchester OWL, the participants had difficulty in finding their way among multiple parentheses and complex disjunctions. With model outlines, they apparently thought that the proposed situation had to satisfy *all* cases.

For the family relations domain, we note the following highlights:

Question 6 elicited *4 errors* using Manchester OWL, and *no errors* using model outlines. This question asked if a person satisfying the given specification could have jobless children. The specification in Manchester OWL included the sentence "*hasChild* **SOME** (*Man* **AND** *worksAt* **ONLY** *Hospital*)". Apparently, the users forgot that "ONLY" (which stands for "∀") does not imply the existence of objects. In the model outline, the presence of a dashed cluster, a dashed box and a dashed arrow made it clear that existence was not required.

Question 8 elicited *3 errors* using Manchester OWL, and *1 error* using model outlines. This question asked if a person satisfying the given specification had to have a grandchild working as a surgeon. Some users found it confusing to follow the composition of roles (*hasChild*–*hasChild*), and were again, as in question 8 about graduate courses, confused by the Manchester OWL abbreviation "**SOME** [· · ·]". In the model outline, the presence of a solid cluster labeled *Surgeon* inside an "among which" inner box made the correct answer more clear.

Domain and type of question	Correct answers (model outlines)	Correct answers (Manchester OWL)
Family, no cases	94%	72%
Courses, no cases	96%	84%
Family, with cases	84%	72%
Courses, with cases	84%	80%

Table 2. Number of correct answers per domain and type of question

One trend was clearly observed in both domains: specifications that involve cases (i.e., complex disjunctions), such as the one in Fig. 5 of this paper, are more difficult to understand than those that do not, as Table 2 indicates.

Among the comments offered by the participants, many indicated confusion due to the way cases were presented in model outlines (like in Fig. 5 of this paper, the layout consisted of 4 diagrams on a single page). Some users thought that all 4 diagrams had to be satisfied. This is clearly one weakness of model outlines (on paper) that we must try to eliminate in the GUI implementation. We predict that such confusion will not arise if the user interacts with the model outline (e.g., dynamically expanding and collapsing cases). The GUI should also make clear when clusters in different cases actually correspond to the same cluster, by showing one single cluster which can be expanded in different ways.

As for time: in the courses domain, each user took in average 28 seconds per question, regardless of the notation. In the family relations domain, each user took in average 26 seconds per question with model outlines, but 40 seconds per question with Manchester OWL.

Of the 10 participants, 5 said they preferred model outlines, 4 said they liked both notations equally well, and 1 said both notations were equally bad.

6 Conclusions

The main achievements of the work related here are the reformulation of our model outline notation and the results of our first usability test, comparing model outlines to Manchester OWL.

Ontology visualization is a very active field of study. The survey [6] discusses over 40 ontology visualization tools, all of them developed in the past 10 years. All of those tools are *general*, in the sense that they use one single visualization framework to show several types of information about the ontology: the subsumption hierarchy, roles, etc. In particular, those tools show concept descriptions either textually (e.g., Protégé) or in the form of abstract syntax trees (as in Fig. 1 of this paper).

Model outlines, on the other hand, are *specialized*, having been designed specifically to show concept descriptions. Although the notation used is new, our usability test indicates it is intuitive enough to be understood by nonspecialists. The specialized nature of model outlines suggests that they can be *integrated*

with a more general tool, so that users can easily switch views, e.g., from the subsumption hierarchy as a tree to the definition of a class as a model outline.

We are currently implementing a concept description browser based on model outlines, as a Protégé plugin. We are taking special care to rely on graphical conventions and interaction techniques that profit from the vast body of knowledge related to visual perception and cognitive principles, as described, e.g., in [15].

Work is also under way to extend model outlines to the concept language associated to OWL 2 [13].

References

1. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P., eds.: The Description Logic Handbook. 2nd edn. Cambridge University Press (2007)
2. McGuinness, D.L., da Silva, P.P.: Explaining answers from the semantic web: the inference web approach. Journal of Web Semantics **1**(4) (2004) 397–413
3. Bailey, J., Bry, F., Furche, T., Schaffert, S.: Web and semantic web query languages: A survey. In Eisinger, N., Maluszynski, J., eds.: Reasoning Web. Volume 3564 of Lecture Notes in Computer Science., Springer (2005) 35–133
4. Horridge, M., Drummond, N., Goodwin, J., Rector, A., Stevens, R., Wan, H.: The Manchester OWL syntax. In: OWL: Experiences and Directions. (2006)
5. Fuchs, N.E., Kaljurand, K., Schneider, G.: Attempto Controlled English meets the challenges of knowledge representation, reasoning, interoperability and user interfaces. In: FLAIRS 2006. (2006)
6. Katifori, A., Halatsis, C., Lepouras, G., Vassilakis, C., Giannopoulou, E.: Ontology visualization methods—a survey. ACM Comput. Surv. **39**(4) (2007) Article 10
7. Gaines, B.R.: Designing visual languages for description logics. Journal of Logic, Language and Information **18**(2) (2009) 217–250
8. Borgida, A., Franconi, E., Horrocks, I., McGuinness, D.L., Patel-Schneider, P.F.: Explaining ALC subsumption. In: International Workshop on Description Logics (DL'99), Linköping, Sweden. (1999)
9. Krivov, S., Williams, R., Villa, F.: GrOWL: A tool for visualization and editing of OWL ontologies. Journal of Web Semantics **5**(2) (2007) 54–57
10. do Amaral, F.N., Bazílio, C.: Visualization of Description Logic models. In: The 21st International Workshop on Description Logics (DL2008), Dresden, Germany. (2008) Available at `http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-353/NaufelMartins.pdf`.
11. do Amaral, F.N.: Visualizing the semantics (not the syntax) of concept descriptions. In: Proceedings of VI TIL, Vila Velha, ES. (2008) Available at `http://www.nilc.icmc.usp.br/til/til2008/p336-do_amaral.pdf`.
12. Howse, J., Molina, F., Shin, S.J., Taylor, J.: On diagram tokens and types. In: DIAGRAMS '02: Proceedings of the Second International Conference on Diagrammatic Representation and Inference, London, UK, Springer-Verlag (2002) 146–160
13. Horrocks, I., Kutz, O., Sattler, U.: The even more irresistible $\mathcal{SROIQ}$. In: Proc. of the 10th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR2006), AAAI Press (2006) 57–67
14. Dumas, J., Redish, J.: A Practical Guide to Usability Testing. Intellect (1994)
15. Ware, C.: Information Visualization: Perception for Design. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2004)