The Graphol Language for
OWL 2 Ontology Editing and Visualization

Domenico Lembo, Valerio Santarelli, Domenico Fabio Savo

Dipartimento di Ingegneria Informatica, Automatica e Gestionale
Sapienza, Università di Roma, Italy

Abstract

In this paper we study Graphol, a fully graphical language inspired by standard languages for conceptual modeling, like the UML class diagram and the ER diagrams, but equipped with a formal semantics based on OWL 2. We formally prove that Graphol is equivalent to OWL 2, i.e., it can capture every OWL 2 ontology and viceversa. We then report on successful experiences on adopting Graphol in several industrial projects. Finally we present some usability studies indicating that Graphol is suitable for quick adoption by conceptual modelers familiar with UML and ER.

1. Introduction

There is a long tradition in many areas of computer science of conceptualizing domains of interest in terms of classes and relationships using a graphical or diagrammatic model. Consider, for example, ER (entity-relationship) diagrams[1], ubiquitously used in databases, or UML class diagrams, the de facto standard in software engineering for information modeling (when used as conceptual models rather than to represent software components). While often such diagrams are used in a semi formal way to help communication, it is well-recognized that having a precise semantics is actually needed to avoid ambiguities in design.

Interestingly, the very first conceptual languages developed in AI were also graphical, most prominently semantic networks [2]. However, most work on knowledge representation in AI has focused more on automated reasoning, and has gradually abandoned the graphical conceptual languages in favor of logical languages. This process has started with the famous paper “What’s in a link” [3], which questioned the inherent ambiguity of graphical conceptual languages of the time, and has continued with the work on KL-ONE [4], then followed by the introduction of modern Description Logics (DLs) [5]. Nevertheless, by the early 90’s a research program started to emerge: not disregard, but try to
logically reconstruct graphical conceptual models used in many fields in order to enable automated reasoning on them [6, 7, 8]. This program has actually been one of the thrusts towards more and more expressive DLs that ultimately led to the development of OWL and OWL 2 [9]. Currently, there is a lively research area working to exploit DLs to represent and enable automated reasoning in graphical conceptual models used in software development and information systems [10, 11, 12, 13, 14, 15].

In this paper we bring about the ultimate contribution to this program: we study a graphical formalism, called Graphol, which resembles ER and UML class diagrams, but has an inherent formal semantics based on DLs and is able to fully capture OWL 2.

Graphol comes after a few years of experience in ontology modeling in IT organizations that are knowledgeable on information systems and software engineering, so are familiar with UML and ER, but have only a technological view of ontology languages such as OWL 2. In these contexts, people often struggle to effectively use the logical formalisms through which ontologies are typically specified, thus slowing down the adoption of semantic technologies. Graphol mitigates this problem, since it provides IT people with a formalism for specifying and reading ontologies rooted in conceptual modeling languages they are used to. Indeed, its usage has helped substantially in taking up semantic technologies in industrial use cases.

To get an idea of Graphol and its relation with UML and ER, in Fig. 1 we model a simple situation about students and courses they attend, using a UML class diagram, a Graphol diagram, and a set of DL axioms, all expressible in OWL 2. In all the three versions of the model, we use the same alphabet for predicates. The model says that a student must attend at least a course, that university students (resp. graduate courses) are students (resp. courses), and that university students can only attend graduate courses. One can see that Graphol’s representation of concepts is through rectangles, analogously to UML, while, as in ER, diamonds are used for roles. Furthermore, solid directed arrows represent inclusions, as in UML and ER (cf. the inclusion

---

We assume that the reader is familiar with UML class diagrams, [16], DLs [17] and OWL 2 [9]. Furthermore, we use DL terminology for ontology predicates, e.g., we use “concept” to denote a set of objects, and “role” to denote a binary relationship between them.
between UniversityStudent and Student). However, differently from UML and ER, in GRAPHOL they do not need to involve only named concepts. The use of a diamond, i.e., a “node” in the diagram, allows us to depict, in a simple graphical way, concept expressions over the domain and the range of a role, by connecting it to possibly labeled blank and solid boxes, respectively. For example, in Fig. 1 the blank box labeled with “\textit{forall}” and linked to the role \textit{attends} denotes the set of individuals that attend only graduate courses. The inclusion drawn between UniversityStudent and this concept expression specifies that a university student can only attend graduate courses, which corresponds to the DL inclusion axiom UniversityStudent \sqsubseteq \forall \textit{attends}.GraduateCourse. We remark that this property is not directly expressible in a graphical way in UML, where we need to specify it as an external constraint (cf. the note in the diagram), possibly expressed in a logical language such as OCL.

We notice that the idea of extending or adapting UML or ER to capture OWL is indeed not new. However, GRAPhOL is distinguished from the other proposals with a precise semantics, such as \cite{[18, 19, 20, 21, 22]}, by its ability to capture any OWL 2 ontology in a completely graphical way. Indeed, previous UML-inspired approaches typically require to annotate diagrams with formulas corresponding to complex OWL expressions. Clearly, this hinders both the diagrammatic representation of the ontology and its intuitive understanding. Similar comprehension difficulties may arise, especially in the enterprise context, with other notations for visual ontology editing that are not based on conceptual modeling languages as UML and ER. For example, in the tool GrOWL \cite{[23]}, ontologies are represented through labeled graphs using quite a large number of graphical symbols and labels. In fact, this tool is tailored to OWL 1, and its development has been discontinued to date. In \cite{[24, 25]}, the authors study the relationship between DLs and diagrammatic reasoning systems. However, the correspondence with such notations and the standard OWL 2 still needs further investigation. We also point out that, besides attempts to provide languages for the graphical specification of ontologies, a lot of research work has been dedicated to the issue of ontology visualization (see, e.g., \cite{[26, 27]}). The aim of ontology visualization tools is to ease navigation of the ontology, and thus typically they are mainly focused on providing advanced layout functionalities and mechanisms to extract possibly summarized views of the ontology. The visualization formalisms they use are therefore not thought also for editing tasks.

The rest of this paper is organized as follows. In Section \ref{sec:preliminaries} we provide some preliminaries on DLs. In Section \ref{sec:syntaxSemantics} we give the formal syntax and semantics of the language, and prove its equivalence with OWL 2, i.e., we show that every GRAPHOL diagram corresponds to an OWL 2 ontology, and, conversely, that every OWL 2 ontology corresponds a GRAPHOL diagram. Then, in Section \ref{sec:applications} we report on successful experiences of adoption of GRAPHOL in several industrial projects as the main conceptual modeling and ontology language. In Section \ref{sec:userEvaluations} we provide some user evaluations of the language which show that it can be adopted by non-expert modelers introducing only a minimal overhead with respect to standard conceptual languages such as UML and ER, and that this overhead pays off when the UML/ER diagrams need logical annotation to fully
cure the ontology of interest. More interestingly, these studies show that
GRAPHOL can be adopted by expert conceptual modelers with ease, greatly
facilitating the adoption of a full fledged ontology language, like OWL 2, as a
fully formal conceptual modeling language.

2. Preliminaries

Description Logics (DLs) \[58\] are a family of knowledge representation lan-
guages that can be used to represent the knowledge of a domain of interest in a
structured and formally well-understood way. Description Logics represent the
domain of interest in terms of objects, i.e., individuals, concepts, which are ab-
tractions for sets of objects, and roles, which denote binary relations between
objects. In addition, some DLs distinguish concepts from value-domains, which
denote sets of values, and roles from attributes, which denote binary relations
between object and values.

We consider an overall alphabet \(\Gamma\) partitioned into \(\Gamma_P\) and \(\Gamma_C\). The former is
in turn partitioned into sets of symbols for atomic concepts, atomic roles, atomic
attributes, and atomic value-domains. The latter instead contains symbols for
individual (object and value) constants. Since the DLs that we consider in this
work distinguish between object and value constants, we partition the set \(\Gamma_C\)
into the sets \(\Gamma_O\), which is the set of constants denoting objects, and \(\Gamma_V\), which
is the set of constants denoting values.

Complex concept expressions are constructed starting from atomic concepts
by applying suitable operators. Analogously for complex role and complex at-
ttribute expressions.

We now introduce syntax and semantics of concept, role, attribute, and
value-domain expressions that are of interest in this work. The syntax of DL
expressions is defined by the following rules:

\[
\begin{align*}
  C & \rightarrow A \mid \neg C \mid C \sqcap \cdots \sqcap C \mid C \sqcup \cdots \sqcup C \mid \exists R \mid \exists R.C \mid \forall R.C \\
  & \mid \geq n R.C \mid \leq n R.C \mid \exists R:\text{Self} \mid \exists V.F \mid \forall V.F \mid \\
  & \mid \geq n V.F \mid \leq n V.F \mid \{c_1, \ldots, c_n\} \mid \top_C \mid \bot_C \\
  F & \rightarrow T \mid \exists V^\neg \mid \neg F \mid F \sqcap \cdots \sqcap F \mid F \sqcup \cdots \sqcup F \\
  & \mid \top_D \mid \bot_D \mid \{w_1, \ldots, w_n\} \\
  R & \rightarrow P \mid P^\neg \mid \neg R \mid R \circ R \mid \top_R \mid \bot_R \\
  V & \rightarrow U \mid \neg V \mid \top_A \mid \bot_A
\end{align*}
\]

where \(A\) denotes an atomic concept, \(P\) an atomic role, \(U\) an atomic attribute
(or simply attribute), \(T\) an atomic value-domain, \(\top_C\) (resp., \(\top_R, \top_A, \top_D\)) the
universal concept (resp., role, attribute, value-domain), and \(\bot_C\) (resp., \(\bot_R, \bot_A, \bot_D\)) the empty concept (resp., role, attribute, value-domain).

We call \(C\) an arbitrary concept, \(R\) an arbitrary role, \(V\) an arbitrary attribute,
and \(F\) an arbitrary value-domain. All these symbols will be used with subscripts,
when needed. \(P^\neg\) denotes the inverse of an atomic role, while \(\neg C, \neg R, \neg V, \neg F\) denote respectively the negation of an arbitrary concept \(C\), an arbitrary role
\(R\), an arbitrary attribute \(V\), and an arbitrary value-domain \(F\). \(\exists V^\neg\) denotes
the range of the attribute $V$. The concept $\exists R$, also called unqualified existential restriction, denotes the domain of a role $R$, i.e., the set of objects that $R$ relates to some object. Similarly, $\exists V$ denotes the domain of an attribute $V$, i.e., the set of objects that $V$ relates to some value. The concept $\exists R.C$, also called qualified existential restriction, denotes the qualified domain of $R$ with respect to the filler $C$, i.e., the set of objects that $R$ relates to some instance of $C$. Similarly, $\exists V.F$ denotes the qualified domain of $V$ with respect to a value-domain $F$, i.e., the set of objects that $V$ relates to some value in $F$. We refer to the qualified existential restriction expression $\exists R_1, \ldots, R_n.C$, where $C$ is not a qualified existential restriction, as an existential role chain of depth $n$.

The concept $\forall R.C$, also called qualified universal restriction, denotes the set of objects for which if there exists an object that $R$ relates them to, then that object is an instance of $C$. Similarly, $\forall V.F$ denotes the set of objects for which if there exists a value that $V$ relates them to, then that value is in $F$. The concept $\exists R.Self$ is used to express local reflexivity to a role $R$. $\exists V^-$ denotes the range of an attribute $V$, i.e., the set of values to which $V$ relates some object. Note that the range $\exists V^-$ of $V$ is a value-domain, whereas the domain $\exists V$ of $V$ is a concept. $\{c_1, \ldots, c_n\}$ denotes the concept which represents the set consisting of the individuals $c_1, \ldots, c_n$, and, similarly, $\{w_1, \ldots, w_n\}$ denotes the value-domain which represents the set consisting of the values $w_1, \ldots, w_n$. $\cap$ and $\cup$ are the usual AND and OR logical connectives. $\leq$ and $\geq$ indicate number restrictions, respectively at-most restriction and at-least restriction, where $n$ ranges over the nonnegative integers. Finally, $R \circ R$ denotes a role chain.

The semantics of a DL KB is given in terms of First-Order Logic (FOL) interpretations. An interpretation $I = (\Delta^I, \cdot^I)$ consists of a nonempty interpretation domain $\Delta^I$, which is the disjoint union of two non-empty sets: $\Delta^I_0$, called the domain of objects, and $\Delta^I_1$, called the domain of values, and an interpretation function, i.e., a function that assigns an element of $\Delta^I$ to each constant in $\Gamma$, a subset of $\Delta^I$ to each concept and value-domain, and a subset of $\Delta^I \times \Delta^I$ to each role and attribute.

The semantics of the constructs is shown in Table 1 illustrating the rules for the interpretation function $\cdot^I$. In the table $c^I$ and $w^I$ (possibly with subscripts) denote the interpretation of object and value constants, i.e., $c^I \in \Delta^I_0$ and $w^I \in \Delta^I_1$.

Given a DL language $\mathcal{L}$, an $\mathcal{L}$-KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ over the alphabet $\Gamma$ is a pair formed by a finite set $\mathcal{T}$ of intensional assertions, i.e., axioms specifying general properties of concepts, roles, and relations, expressed in $\mathcal{L}$, called TBox, and a finite set $\mathcal{A}$ of extensional assertions, i.e., axioms about individual objects, over $\Gamma$ expressed in $\mathcal{L}$, called ABox. In this work we focus on techniques for the representation of the TBox, so we will disregard the ABox.

The TBox assertions we focus on in this work are:

\[
\begin{align*}
C_1 & \sqsubseteq C_2 & \text{(concept inclusion)}; \\
R_1 & \sqsubseteq R_1 & \text{(role inclusion)}; \\
V_1 & \sqsubseteq V_2 & \text{(attribute inclusion)}; \\
F_1 & \sqsubseteq F_2 & \text{(value-domain inclusion)}. \\
\end{align*}
\]
<table>
<thead>
<tr>
<th>Construct</th>
<th>Syntax</th>
<th>Semantics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Atomic concept</td>
<td>$A$</td>
<td>$A^I \subseteq \Delta^I_0$</td>
</tr>
<tr>
<td>Atomic role</td>
<td>$P$</td>
<td>$P^I \subseteq \Delta^I_0 \times \Delta^I_0$</td>
</tr>
<tr>
<td>Atomic attribute</td>
<td>$U$</td>
<td>$U^I \subseteq \Delta^I_0 \times \Delta^I_0$</td>
</tr>
<tr>
<td>Atomic value-domain</td>
<td>$T$</td>
<td>$(^I T) \subseteq \Delta^I_0$</td>
</tr>
<tr>
<td>Universal concept</td>
<td>$\top_C$</td>
<td>$\Delta^I_0$</td>
</tr>
<tr>
<td>Universal role</td>
<td>$\top_R$</td>
<td>$\Delta^I_0 \times \Delta^I_0$</td>
</tr>
<tr>
<td>Universal attribute</td>
<td>$\top_A$</td>
<td>$\Delta^I_0 \times \Delta^I_0$</td>
</tr>
<tr>
<td>Universal value-domain</td>
<td>$\top_D$</td>
<td>$\Delta^I_0$</td>
</tr>
<tr>
<td>Empty concept, role</td>
<td>$\bot_C, \bot_R, \bot_A, \bot_D$</td>
<td>$\emptyset$</td>
</tr>
<tr>
<td>attribute, value-domain</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unqualified role</td>
<td>$\exists R$</td>
<td>${ o</td>
</tr>
<tr>
<td>existential restriction</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Qualified role</td>
<td>$\exists R.C$</td>
<td>${ o</td>
</tr>
<tr>
<td>existential restriction</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Qualified role</td>
<td>$\forall R.C$</td>
<td>${ o</td>
</tr>
<tr>
<td>universal restriction</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Qualified maximum</td>
<td>$\leq n R.C$</td>
<td>${ o</td>
</tr>
<tr>
<td>cardinality role</td>
<td></td>
<td></td>
</tr>
<tr>
<td>restriction</td>
<td>$\geq n R.C$</td>
<td>${ o</td>
</tr>
<tr>
<td>Self restriction</td>
<td>$\exists R,Self$</td>
<td>${ o</td>
</tr>
<tr>
<td>Unqualified attribute</td>
<td>$\exists V$</td>
<td>${ o</td>
</tr>
<tr>
<td>existential restriction</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Qualified attribute</td>
<td>$\exists V.F$</td>
<td>${ o</td>
</tr>
<tr>
<td>existential restriction</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Qualified attribute</td>
<td>$\forall V.F$</td>
<td>${ o</td>
</tr>
<tr>
<td>universal restriction</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Qualified maximum</td>
<td>$\leq n V.F$</td>
<td>${ o</td>
</tr>
<tr>
<td>cardinality attribute</td>
<td></td>
<td></td>
</tr>
<tr>
<td>restriction</td>
<td>$\geq n V.F$</td>
<td>${ o</td>
</tr>
<tr>
<td>Qualified minimum</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cardinality attribute</td>
<td></td>
<td></td>
</tr>
<tr>
<td>restriction</td>
<td></td>
<td></td>
</tr>
<tr>
<td>One-of (concept)</td>
<td>${c_1, \ldots, c_n}$</td>
<td>${c_1', \ldots, c_n'}$</td>
</tr>
<tr>
<td>One-of (value-domain)</td>
<td>${w_1, \ldots, w_n}$</td>
<td>${w_1', \ldots, w_n'}$</td>
</tr>
<tr>
<td>Attribute range</td>
<td>$\exists^V^{\neg}$</td>
<td>${ v</td>
</tr>
<tr>
<td>Inverse role</td>
<td>$^P^\neg$</td>
<td>${ (o, o')</td>
</tr>
<tr>
<td>Role chain</td>
<td>$R \circ R$</td>
<td>${ (o, o')</td>
</tr>
<tr>
<td>Concept negation</td>
<td>$\neg C$</td>
<td>$\Delta^I_0 \setminus C^I$</td>
</tr>
<tr>
<td>Role negation</td>
<td>$\neg R$</td>
<td>$(\Delta^I_0 \times \Delta^I_0) \setminus R^I$</td>
</tr>
<tr>
<td>Attribute negation</td>
<td>$\neg V$</td>
<td>$(\Delta^I_0 \times \Delta^I_0) \setminus V^I$</td>
</tr>
<tr>
<td>Value-domain negation</td>
<td>$\neg F$</td>
<td>$(\Delta^I_0) \setminus F^I$</td>
</tr>
<tr>
<td>Concept conjunction</td>
<td>$C_1 \sqcap \cdots \sqcap C_n$</td>
<td>$C_1^I \cap \cdots \cap C_n^I$</td>
</tr>
<tr>
<td>Concept disjunction</td>
<td>$C_1 \sqcup \cdots \sqcup C_n$</td>
<td>$C_1^I \cup \cdots \cup C_n^I$</td>
</tr>
</tbody>
</table>

Table 1: The DL constructs with their semantics.
Concept inclusion assertions state that all instances of one concept are also instances of another concept. Analogously for role, attribute, and value-domain inclusion assertions.

We specify the semantics of the TBox assertions mentioned above by defining when an interpretation $\mathcal{I}$ satisfies an inclusion assertion:

$$C_1 \subseteq C_2 \text{ if } C_1^I \subseteq C_2^I;$$

$$V_1 \subseteq V_2 \text{ if } V_1^I \subseteq V_2^I;$$

$$R_1 \subseteq R_2 \text{ if } R_1^I \subseteq R_2^I;$$

$$F_1 \subseteq F_2 \text{ if } F_1^I \subseteq F_2^I.$$
### Figure 2: Graphol predicate nodes, constructor nodes, and edges.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Name</th>
<th>Symbol</th>
<th>Name</th>
<th>Symbol</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Concept node</td>
<td></td>
<td>Role node</td>
<td></td>
<td>Attribute node</td>
</tr>
<tr>
<td></td>
<td>Value-domain node</td>
<td></td>
<td>Individual/Value node</td>
<td></td>
<td>Domain restriction node</td>
</tr>
<tr>
<td>or</td>
<td>Union node</td>
<td></td>
<td>Intersection node</td>
<td></td>
<td>Range restriction node</td>
</tr>
<tr>
<td>not</td>
<td>Complement node</td>
<td></td>
<td>Inverse node</td>
<td></td>
<td>Inclusion edge</td>
</tr>
<tr>
<td>chain</td>
<td>Role chain node</td>
<td></td>
<td>One-of node</td>
<td></td>
<td>Input edge</td>
</tr>
</tbody>
</table>

which are represented as squares and, informally, are used to construct complex concepts that denote sets of objects that have a particular participation to roles or attributes, and operator nodes, which are represented as hexagons, and correspond to logical operators such as intersection, union, and negation, as well as operators like role inverse, one-of, and role chain. The label on restriction nodes, indicated by “Restriction type” on the domain and range restriction node in Figure 2, can have one of the following values: “exists”, for existential restriction, “forall”, for universal restriction, “self”, for self restriction, and “(x, −)” and “(−, y)”, with x and y positive integers, for minimum and maximum cardinality restrictions, respectively. The label on operator nodes instead is used to identify the kind of operator the node is representing, and its value must be one of those shown in Figure 2 inside the hexagonal shapes.

We observe that Graphol uses three visual variables [59] to encode the predicate and constructor nodes. These are (i) the shape of the nodes, which is used to distinguish between predicate nodes, restriction nodes, and operator nodes, and among the different kinds of predicate nodes; (ii) the size of the nodes, which allows to clearly discriminate between the quadrilaterals used for concept nodes, value-domain nodes, and restriction nodes; and (iii) brightness, which is used to distinguish the domain restriction node from the range restriction node.

To avoid encumbering the user in learning the Graphol syntax, we have chosen to limit the number of different graphical symbols used to depict Graphol nodes, and to maintain it to around seven, which is the commonly recognized ideal upper bound for software engineering graphical languages [60]. Indeed, experimental studies demonstrate that a high number of different symbols in a language for software engineering increases the learning difficulty by non-expert users [61]. Towards this goal, we adopt a single shape, the hexagon, to represent all operator nodes, and adopt textual encoding, by means of the label inside the shape, to distinguish between different types of operator nodes.

In Graphol, an edge can be of two types: input edge or inclusion edge.
The former is a dashed directed diamond edge, where the diamond indicates the target end, and is used to construct an expression in the ontology by linking constructor and predicate nodes. The latter is a solid directed arrow edge, where the arrow indicates the target end, and is used to represent an inclusion assertion in the ontology.

A Graphol expression is a weakly-connected acyclic directed graph, whose nodes are both predicate and constructor nodes and whose edges are only input edges, that has exactly one node with no outgoing edges. We call such node the sink of the expression. We recall that a directed graph is called weakly-connected if replacing all of its directed edges with undirected edges produces a connected (undirected) graph. In the following definition we provide the four kinds of expressions that are admitted in Graphol.

Definition 1. A Graphol expression is built through one of the four following specifications.

1. A concept expression is a weakly-connected acyclic graph, defined inductively as follows.
   - a concept node is a concept expression, whose sink node is the concept itself;
   - a domain or range restriction node, labeled “exists”, “forall”, “(x, -)”, or “(-, y)”, with one input role expression and one input concept expression, is a concept expression, whose sink node is the domain or range restriction node;
   - a domain or range restriction node, labeled “self”, with one input role expression is a concept expression, whose sink node is the domain or range restriction node;
   - a domain restriction node, labeled “exists”, “forall”, “(x, -)”, or “(-, y)”, with one input attribute expression and one input value-domain expression, is a concept expression, whose sink node is the domain restriction node;
   - an intersection node or a union node with $n \geq 2$ input concept expressions is a concept expression, whose sink node is the intersection node or the union node;
   - a complement node with one input concept expression is a concept expression, whose sink node is the complement node;
   - a one-of node with $n \geq 1$ input individual nodes is a concept expression, whose sink node is the one-of node.

2. A role expression is a weakly-connected acyclic graph, defined inductively as follows.
   - a role node is a role expression, whose sink node is the role itself;
   - an inverse node with one input role expression is a role expression, whose sink node is the inverse node;
– a complement node with one input role expression is a role expression, whose sink node is the complement node;

– a chain node with \( n \geq 2 \) input role expressions, each one labeled with \( i \) such that \( 1 \leq i \leq n \) and \( i \neq j \) for any two labels \( i \) and \( j \), is a role expression, whose sink node is the chain node.

3. An attribute expression is a weakly-connected acyclic graph, defined inductively as follows.

– an attribute node is an attribute expression, whose sink node is the attribute itself;

– a complement node with one input attribute expression is an attribute expression, whose sink node is the complement node.

4. A value-domain expression is a weakly-connected acyclic graph, defined inductively as follows.

– a value-domain node is a value-domain expression, whose sink node is the value-domain node itself;

– a range restriction node, labeled “exists”, with one input attribute expression is a value-domain expression, whose sink node is the range restriction node;

– an intersection node or a union node with \( n \geq 2 \) input value-domain expressions is a value-domain expression, whose sink node is the intersection node;

– a complement node with one input value-domain expression is a value-domain expression, whose sink node is the complement node;

– a one-of node with \( n \geq 1 \) input value nodes is a value-domain expression, whose sink node is the one-of node.

As usual in ontology languages and DLs, intentional assertions are specified through inclusions. Then, a Graphol (inclusion) assertion is expressed through an inclusion edge going from the (sink node of the) subsumed expression to the (sink node of the) subsumer. In detail, a concept (resp., role, attribute, value-domain) inclusion assertion is obtained by linking two sink nodes of concept (resp., role, attribute, value-domain) expressions. Inclusion edges between expressions of different types (e.g., a concept expression with a role expression) are not allowed.

Finally, we define a Graphol ontology as a set of Graphol assertions.

In order to ease the task of ontology design and reading through Graphol, we choose to allow the repetition of the same predicate in the ontology through multiple predicate nodes, obviously all labeled with the name of the predicate. This is particularly useful for predicates that occur in numerous assertions in the ontology, whose representation through a single node would therefore produce an abundance of outgoing or incoming edges from that node, and would likely lead to difficult layout issues. It goes without saying that all constructor nodes can also be repeated as many times as necessary.
3.2. Graphol semantics

An important desiderata of visual modeling languages is to have a clear and precise semantics [62, 63]. To this aim, here we provide the semantics of GRAPHOL expressions and assertions by giving their one-to-one correspondence with DL expressions and assertions, which in turn have a formal semantics, as discussed in Section 2.

We start with GRAPHOL expressions, for which we define a function $\Phi$ that takes as input a GRAPHOL expression $E_G$ and returns a DL expression that represents it.

In formalizing $\Phi$, we denote with $\text{sink}(E_G)$ the sink node of $E_G$, and with $\text{arg}(E_G)$ the (possibly empty) set of GRAPHOL expressions that are linked to $\text{sink}(E_G)$ by means of input edges. Then, we define $\Phi$ as follows:

- if $\text{sink}(E_G)$ is a concept, role, attribute, value-domain, or individual/value node labeled $S$, then $\Phi(E_G) = S$;
- if $\text{sink}(E_G)$ is a domain restriction node labeled “exists” (resp., “forall”, “($x$, -)”, “(-, y)”), and $\text{arg}(E_G) = \{\epsilon_{RA}, \epsilon_{CV}\}$, where either $\epsilon_{RA}$ is a GRAPHOL role expression and $\epsilon_{CV}$ is a GRAPHOL concept expression or $\epsilon_{RA}$ is a GRAPHOL attribute expression and $\epsilon_{CV}$ is a GRAPHOL value-domain expression, then $\Phi(E_G) = \exists \Phi(\epsilon_{RA}).\Phi(\epsilon_{CV})$ (resp., $\Phi(E_G) = \forall \Phi(\epsilon_{RA}).\Phi(\epsilon_{CV})$, $\Phi(E_G) \geq x \ \Phi(\epsilon_{RA}).\Phi(\epsilon_{CV})$, $\Phi(E_G) \leq y \ \Phi(\epsilon_{RA}).\Phi(\epsilon_{CV})$);
- if $\text{sink}(E_G)$ is a range restriction node labeled “exists” (resp., “forall”, “($x$, -)”, “(-, y)”), and $\text{arg}(E_G) = \{\epsilon_R, \epsilon_C\}$, where $\epsilon_R$ is a GRAPHOL role expression and $\epsilon_C$ is a GRAPHOL concept expression, then $\Phi(E_G) = \exists(\Phi(\epsilon_R))^{-}.\Phi(\epsilon_C)$ (resp. $\Phi(E_G) = \forall(\Phi(\epsilon_R))^{-}.\Phi(\epsilon_C)$, $\Phi(E_G) \geq x \ (\Phi(\epsilon_R))^{-}.\Phi(\epsilon_C)$, $\Phi(E_G) \leq y \ (\Phi(\epsilon_R))^{-}.\Phi(\epsilon_C)$);
- if $\text{sink}(E_G)$ is a domain (resp., range) restriction node labeled “self”, and $\text{arg}(E_G) = \{\epsilon_{RA}\}$, where $\epsilon_{RA}$ is a GRAPHOL role expression, then $\Phi(E_G) = \exists \Phi(\epsilon_{RA}).\text{Self}$ (resp., $\Phi(E_G) = \exists(\Phi(\epsilon_{RA}))^{-}.\text{Self}$);
- if $\text{sink}(E_G)$ is a range restriction node labeled “exists” and $\text{arg}(E_G) = \{\epsilon_A\}$, where $\epsilon_A$ is a GRAPHOL attribute expression, then $\Phi(E_G) = \exists(\Phi(\epsilon_A))^{-}$;
- if $\text{sink}(E_G)$ is an intersection (resp. a union or a one-of node) and $\text{arg}(E_G) = \{\epsilon^1, \ldots, \epsilon^n\}$, then $\Phi(E_G) = \bigcap_{i=1}^n \Phi(\epsilon^i)$ (resp. $\Phi(E_G) = \bigcup_{i=1}^n \Phi(\epsilon^i)$, $\Phi(E_G) = \{\Phi(\epsilon^1), \ldots, \Phi(\epsilon^n)\}$);
- if $\text{sink}(E_G)$ is a complement node and $\text{arg}(E_G) = \{\epsilon\}$, then $\Phi(E_G) = \neg \Phi(\epsilon)$.\footnote{Obviously, we assume the alphabets for concepts, roles, attributes, value-domains, individuals, and values to be pairwise disjoint, so that it is clear which kind of predicate $S$ represents. Also, with a little abuse of notation, if $S = \text{"Top"}$ (resp. $S = \text{"Bottom"}$), we assume the $\Phi$ returns the correct DL universal (resp. empty) predicate, depending on whether $\text{sink}(E_G)$ is a concept, role, attribute, or value-domain.}
− if sink\((E_G)\) is an inverse node and \(\text{arg}(E_G) = \{\epsilon_R\}\), then \(\Phi(E_G) = (\Phi(\epsilon_R))^{-}\);

− if sink\((E_G)\) is a chain node and \(\text{arg}(E_G) = \{\epsilon^1_R, \ldots, \epsilon^n_R\}\), where each \(\epsilon^i_R\), with \(1 \leq i \leq n\), is a \text{GRAPHOL} role expression linked to sink\((E_G)\) by an input edge labeled \(i\), then \(\Phi(E_G) = \Phi(\epsilon^1_R) \circ \Phi(\epsilon^2_R) \circ \cdots \circ \Phi(\epsilon^n_R)\).

Similarly, in order to provide the semantics of \text{GRAPHOL} inclusion assertions, we define a function \(\Theta\) that takes as input one such inclusion \(\alpha_G\) and returns its DL representation. We denote with \(\text{source}(\alpha_G)\) the \text{GRAPHOL} expression whose sink node is the source of the inclusion edge in \(\alpha_G\), and with \(\text{target}(\alpha_G)\) the \text{GRAPHOL} expression whose sink node is the target of the inclusion edge in \(\alpha_G\). The function \(\Theta\) is thus defined as \(\Theta(\alpha_G) = \Phi(\text{source}(\alpha_G)) \sqsubseteq \Phi(\text{target}(\alpha_G))\).

Given a \text{GRAPHOL} ontology \(O_G\), we are now able to translate it into a DL ontology \(O_{DL}\) by applying \(\Theta\) to each assertion in \(O_G\). Then, the semantics of \(O_G\) coincides with the semantics of \(O_{DL}\), for which we refer to Section 2.

In Tables 2 and 3 we provide an example of the application of the function \(\Phi\) to \text{GRAPHOL} expressions of “depth” 0 or 1, i.e., expressions formed only by predicate nodes or by a constructor node taking as input only predicate nodes.

### 3.3. Shortcuts

In order to aid the user in the design of a \text{GRAPHOL} ontology, we have defined some shortcuts that allow for a more compact representation of frequently used expressions and assertions. Here we introduce them, and provide examples of their use.

The first shortcut is the \textit{disjoint union node}, which is shaped as a black hexagon. This special node is used to represent a union expression, and at the same time it imposes the disjointness between its arguments. Therefore, it also represents negative inclusion assertions involving its arguments. This shortcut is particularly useful for defining disjoint concept hierarchies. In Figure 3 we give an example of one such hierarchy in the two versions, with and without the use of the shortcut.

**Figure 3:** Example of a disjoint concept hierarchy represented in \text{GRAPHOL} with (left-hand side figure) and without (right-hand side figure) the disjoint union node.

The second shortcut we introduce is a compact notation for the definition of the existential domain or range restriction on a role (resp. attribute) taking as
<table>
<thead>
<tr>
<th>Expression</th>
<th>Graphol Syntax</th>
<th>DL Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>Atomic Concept</td>
<td>$\text{C}$</td>
<td>$C$</td>
</tr>
<tr>
<td>Domain restriction on role</td>
<td>$\exists R \cdot C$ $\forall R \cdot C$ $\geq xR.C$ $\leq yR.C$</td>
<td></td>
</tr>
<tr>
<td>Range restriction on role</td>
<td>$\exists R^{-} \cdot C$ $\forall R^{-} \cdot C$ $\geq xR^{-} \cdot C$ $\leq yR^{-} \cdot C$</td>
<td></td>
</tr>
<tr>
<td>Domain restriction on attribute</td>
<td>$\exists V.F$ $\forall V.F$ $\geq xV.F$ $\leq yV.F$</td>
<td></td>
</tr>
<tr>
<td>Concept Intersection</td>
<td>$C_1 \sqcap C_2$</td>
<td>$C_1 \cap C_2$</td>
</tr>
<tr>
<td>Concept Union</td>
<td>$C_1 \sqcup C_2$</td>
<td>$C_1 \cup C_2$</td>
</tr>
<tr>
<td>Concept Complement</td>
<td>$\neg C$</td>
<td></td>
</tr>
<tr>
<td>Concept One-of</td>
<td>${a, b, c}$</td>
<td></td>
</tr>
<tr>
<td>Self Restriction</td>
<td>$\exists R.Self$</td>
<td></td>
</tr>
<tr>
<td>Atomic Role</td>
<td>$P$</td>
<td></td>
</tr>
<tr>
<td>Role Inverse</td>
<td>$R^{-}$</td>
<td></td>
</tr>
<tr>
<td>Role Complement</td>
<td>$\neg R$</td>
<td></td>
</tr>
<tr>
<td>Role Chain</td>
<td>$R_1 \circ R_2$</td>
<td>$R_1 \circ R_2$</td>
</tr>
</tbody>
</table>

Table 2: DL-GRAPHOL correspondence for concept and role expressions of depth 0 or 1. $C$, $C_1$, and $C_2$ denote an atomic concept, and $R$, $R_1$, and $R_2$ denote an atomic role.
Table 3: DL-Graphol correspondence for attribute and value-domain expressions of depth 0 or 1. V denotes an atomic attribute, and F, F₁, and F₂ denote an atomic value-domain.

input the universal concept (resp. universal value-domain), which is equivalent to an unqualified existential restriction. Since this is typically the most recurring restriction, we allow to omit the universal concept (resp. value-domain). In other words, to express the unqualified existential restriction on a role or attribute, one can simply link a role or attribute sub-graph to an existential restriction node, as shown in Figure 4.

Figure 4: Example of an existential restriction in Graphol with (left-hand side figure) and without (right-hand side figure) the compact notation.

A commonly used assertion in ontologies is the one which specifies that a role R (resp., an attribute V) is globally functional, i.e., $\top C \sqsubseteq \leq 1 R \top C$. 
Therefore, in order to provide a more compact representation of the assertion in Graphol, we indicate that a role is functional (resp. inverse functional) by means of a diamond with an empty double border (resp. solid double border). Analogously, for a functional attribute. In case of a role that is both functional and inverse functional, its nodes will have a split empty and solid double border. As an example, on the left in Figure 5 we show the standard Graphol assertion that defines the functionality of a role, and on the right hand side, the compact notation for a functional role (top left), a inverse functional role (top right), a role that is both functional and inverse functional (bottom left), and, finally, a functional attribute (bottom right).

Figure 5: Example of a global functional role represented without the compact notation (left-hand side figure), and functional role, inverse functional role, functional and inverse functional role, and functional attribute (resp., top left, top right, bottom left, and bottom right) represented with the compact notation.

We further notice that two domain (resp. range) restriction nodes with labels \((x,-)\) and \((-,y)\), respectively, and the same input expressions can be substituted by a single domain (resp. range) restriction node with label \((x,y)\) and the same inputs. In other terms, min and max cardinality restrictions can be drawn together. Lastly, if one wishes to define equivalence between two expressions of the same kind, it is possible to use a single inclusion edge with an arrow both on the source end and on the target end instead of two inclusion edges (with the arrow only on the target end) in opposite directions.

3.4. Graphol and OWL 2

In this section we study the relationship between Graphol and OWL 2 [64], the W3C standard ontology language. We first study whether Graphol ontologies can be entirely expressed in OWL 2, and then consider the other way around. According to the Graphol syntax given earlier, Graphol expressiveness goes slightly beyond that of OWL 2. This allowed us to maintain the formal definition of the syntax of our language simple, without burdening it with too many syntactic categories. Of course, due to this choice, reasoning in full Graphol is undecidable [17]. However, by suitably restricting the way in which Graphol expressions can be combined, we easily get decidable languages. In particular, we can limit Graphol in such a way that it becomes translatable in OWL 2. To precisely describe this restriction, we need to define basic role expressions, which are expressions constituted by either a role node
or the inverse node with a role node as input. Below we give a proviso needed to our aims.

**Proviso.** Role expressions given as input to domain or range restriction nodes, to self nodes, to inverse nodes, to complement nodes, or to chain nodes can be only basic role expressions. Attribute expressions given as input to domain restriction nodes can be only attribute nodes. Role and attribute expressions constructed with complement nodes cannot be the source of any inclusion edge. Role expressions constructed with chain nodes cannot be the target of any inclusion edge and can be included only in basic role expressions. Attribute expressions in input to range restriction nodes can be only attribute nodes. Graphol inclusion assertions between value-domain expressions must involve at least a value-domain node (i.e., the source or the target of the assertion must be an atomic data type). Finally, the ontology must obey to the same global restrictions imposed on OWL 2 (e.g., only regular role hierarchies are allowed and only simple roles can occur in cardinality restrictions) \[65, 64\].

**Theorem 1.** Every Graphol ontology constructed under the above proviso is correctly translatable into an OWL 2 TBox in linear-time.

*Proof (sketch).* The function \( \Theta \) given in Subsection 3.2 is obviously applicable to a Graphol ontology restricted according to the above proviso. It is then easy to verify that each DL inclusion assertion returned by \( \Theta \) is an OWL 2 axiom. Since \( \Theta \) returns one DL inclusion assertion for each Graphol inclusion assertion, the cost of the transformation is linear. \( \square \)

The following theorem considers instead transformation of OWL 2 TBoxes in Graphol.

**Theorem 2.** Every OWL 2 TBox is correctly translatable into a Graphol ontology in linear-time.

*Proof (sketch).* We first observe that any OWL 2 ontology can be written as a set of inclusion assertions. Below we provide some well-known correspondences useful for this aim (\( Q \) denotes a role or its inverse, whereas \( \top_C \) and \( \top_R \) denote the universal concept and role, respectively).

\[
\begin{align*}
\text{(funct } Q) & \iff \top_C \subseteq \leq 1Q.\top_C \\
\text{(symmetric } Q) & \iff Q \subseteq Q^- \\
\text{(reflexive } Q) & \iff \top_R \subseteq \exists Q.\text{Self} \\
\text{(transitive } Q) & \iff Q \circ Q \subseteq Q \\
\text{(asymmetric } Q) & \iff Q \subseteq \neg Q^- \\
\text{(irreflexive } Q) & \iff \top_R \subseteq \neg \exists Q.\text{Self}
\end{align*}
\]

It is then possible to define a function that translates such a normalized TBox in Graphol. Intuitively, this function inverts the function \( \Phi^{-1} \) introduced in Subsection 3.2. More precisely, we first define a function \( \Phi^{-1} \) by induction on the structure of OWL 2 formulas, which can be seen as the inverse of the function \( \Phi \) introduced in Subsection 3.2. For the base case, let \( S \) be an atomic concept (resp., role, attribute, or value-domain), \( \Phi^{-1}(S) \) returns a Graphol concept node (resp. role, attribute, or value-domain node) labeled with \( S \). Let then \( \exists R.A \) (resp. \( \exists R^{-}.A \)) an OWL 2 expression, with \( R \) an atomic role and \( A \) an atomic concept, \( \Phi^{-1}(\exists R.A) \) (resp. \( \Phi^{-1}(\exists R^{-}.A) \)) returns the Graphol...
expression whose sink node is a domain (resp. range) restriction node labeled “exists” taking as input \( \Phi^{-1}(A) \) and \( \Phi^{-1}(R) \). Other inductive cases can be defined analogously. Then, let \( \alpha = \alpha_\ell \sqsubseteq \alpha_r \) be an OWL 2 inclusion assertion, \( \Theta^{-1}(\alpha) = \alpha_G \), where \( \alpha_G \) is the GRAPHOL inclusion assertion whose sink node is \( \Phi^{-1}(\alpha_\ell) \) and target node is \( \Phi^{-1}(\alpha_r) \). It is easy to see that the cost of this transformation is linear.

3.5. Example

In Figure 6, we provide the GRAPHOL specification of a portion of the Pizza ontology we have used throughout this section. We also present below a logically equivalent representation of such ontology given in terms of DL assertions.

![Figure 6: GRAPHOL ontology example: an excerpt of the Pizza ontology.](image)

\[
\begin{align*}
\text{Pizza} & \sqcup \text{PizzaTopping} \sqsubseteq \text{Food} & \text{Pizza} & \sqsubseteq \neg \text{PizzaTopping} \\
\text{MeatTopping} & \sqsubseteq \neg \text{CheeseTopping} & \text{MeatyPizza} & \sqsubseteq \neg \text{VegetarianPizza} \\
\exists \text{hasIngredient} & \sqsubseteq \text{Food} & \exists \text{hasIngredient}^{-} & \sqsubseteq \text{Food} \\
\text{hasTopping} & \sqsubseteq \text{hasIngredient} & \exists \text{hasTopping}^{-} \sqsubseteq \text{PizzaTopping} \\
\exists \text{calories} & \equiv \text{Food} & \exists \text{calories}^{-} \sqsubseteq \text{xsd:integer} \\
(funct \text{calories})
\end{align*}
\]
4. Graphol in practice

Graphol has been and is currently being used in various industrial and academic projects by teams of ontology designers for the specification and maintenance of domain ontologies (see, e.g., [66]). As already discussed, these projects have actually been a source of inspiration for Graphol. At the same time, they have shown Graphol’s validity as a language for ontology design, as it has played a crucial role in their successful accomplishment.

Many of the projects we were involved in share the common aim of realizing Ontology-based Data Access (OBDA) applications. The goal of OBDA is to enable the access and integration of data stored in actual operational databases through the mediation of an ontology, which is connected to such databases through declarative mappings [67]. The main service in OBDA is query answering: users directly pose queries over the ontology and obtain answers computed on the basis of the ontology, the mappings, and the data at the sources. It is well-known that for query answering to be feasible in OBDA the ontology has to be specified in some light DL, like the ones of the DL-Lite family [68]. However, in ontology-based applications it is of paramount importance to refer to an ontology that captures the domain faithfully, in a language that is expressive enough for that purpose. Thus, it is usual to distinguish between a reference, possibly expressive, ontology, and an operational ontology, adapted for specific application purposes from the reference one to take computational issues into account [69]. In our OBDA approach, during the ontology design phase, we thus start by using the expressiveness of the entire OWL language, and then for query answering purposes we automatically extract from this reference ontology a DL-Lite ontology, according to semantic approximation algorithms as those described in [70]. Notice also that the reference ontology is the basis for the development of various services, of which query answering is only an example, towards the realization of a comprehensive semantic approach to information management, which is indeed called Ontology-based Data Management [71].

When we model the ontology, we usually do not look initially at the data sources, but concentrate on domain aspects. This is intended to make the ontology design not affected by the way data are organized in the information systems, where data mainly serve software applications, and their shape does not necessarily reflect the conceptualization of the domain. In this phase, we work hand in hand with domain experts, and the role of Graphol is thus extremely important. It allows us to easily draw initial, skeleton schemas, as in usual top-down strategies for conceptual design. Such schemas are the basis for further discussions and analysis, and are then iteratively refined until they reach a consolidated form. By virtue of the use of Graphol, we are able to easily adopt and explain some classical transformation primitives for conceptual modeling design [72]. Also, we can easily maintain trace of the evolution of the design. These activities, as well as communication with domain experts, would be much more difficult to carry on by using a non-graphical environment for ontology development.

To evaluate whether the realized ontology meets the requirements for its
intended usage, we regularly check during the various modeling phases whether it is possible to specify competency questions over it, i.e., questions that the ontology must be able to answer. These questions are formulated by the domain experts and are a crucial part of the requirements. Graphol turned out to be fundamental to help domain experts to formulate such queries over the ontology, since the diagram naturally allows them to trace the path to be traversed by the queries on the ontology graph. Besides competency questions, this is obviously a characteristic that helps also the definition of generic queries over the ontology at run-time.

So far we have underlined the importance of Graphol in our practical experiences as a mechanism for ontology design, and for obtaining high-quality ontologies. Particularly, this high quality can be perceived in two of the fundamental quality requirements for ontologies [69], i.e., intelligibility, which refers to the understandability of the ontology by humans, and fitness, which refers to its ability of modeling the requirements for its usage. In what follows we give an overview of some of the projects we were involved in, and highlight how the Graphol language evolved through them.

The groundwork for the development of Graphol was laid out during a project commissioned to us by the Monte dei Paschi di Siena (MPS) banking institute with the purpose of experimenting OBDA techniques over the actual MPS data repositories [73]. This was one of our first experiences with business partners, and the formalism that we chose to communicate with domain experts was an enriched version of the ER model, which would then later evolve into Graphol. During this project we actually maintained two versions of the ontology, the ER-based one and the OWL one, where the former was an approximated view of the latter. Evidently, maintaining the two versions aligned was time-consuming and error prone, thus, after this experience, we started the definition of a graphical, ER-based language for ontology specification.

Later on, we were involved in a joint collaboration with the Department of Treasury of the Italian Ministry of Economy and Finance [66, 74]. The focus of the project was on the public debt composition, namely the state liabilities and assets, and the financial instruments used by the Italian public administrations to manage the public debt. This was the first project in which the ontology was specified through (a first version of) Graphol. In this version we used different graphical symbols for the operator nodes, and thus the language had many more graphical elements than the current version. We experienced that the high number of symbols slowed down the learning of Graphol by people of the Ministry, and thus we decided to move towards the new version of the language, in which we adopt only the hexagon for operator nodes, and distinguish among the various operators through labels. As said, such modification makes the language compliant with guidelines for software engineering graphical languages with respect to the used number of symbols [60]. The ontology produced in the

project consists of 359 concepts, 157 roles, 205 attributes, and 2738 assertions. Remarkably, this ontology has been included as a formal specification of the data requirements in a call for tenders for the realization of a new information system for the management of data and processes relative to the public debt domain. Furthermore, the GRAPHOL diagrams have been incorporated into the Mastro Studio system for the documentation and navigation of the ontology, currently in use at the Ministry [75].

We have been then heavily involved in a project with Telecom Italia, the leading Italian telecommunications company, in which we focused on the domain of Telecom Italia’s Access Network. The Access Network constitutes the portion of the Telecom Italia telecommunications network which connects subscribers to the operating centers of their service provider. This project is positioned inside the Telecom Italia Dynamic Inventory scenario, which proposes to create a new and unique platform for Telecom Italia’s network inventory, and its goal is to demonstrate how the OBDA paradigm can be of assistance for the experts of the Access Network in analyzing and exploring the data stored in their repositories. The design and evolution of the ontology has been performed using the most recent version of the GRAPHOL language. The use of GRAPHOL proved to be particularly effective in this scenario, since there are various elements of the Access Network which are extremely technical in nature, and the knowledge on their characteristics often do not cross the boundaries of specific departments within the organization. The ontology has therefore acquired a great importance as a centralized source of knowledge on the company domain, and the use of GRAPHOL allowed us to make it accessible to many different users, none of which had any prior knowledge or experience with ontologies. The current version of the ontology consists in 248 concepts, 93 roles, 132 attributes, and 1679 assertions.

In parallel to the project with Telecom Italia, GRAPHOL has been used in a collaboration with Bloomberg LP. The ontology developed in this project is essentially focused on production, storage, and overseas shipment of petrochemical products, a domain that is of interest for various applications within the Bloomberg information systems. OBDA in this scenario has been tested through this ontology connected to internal, third party, and publicly available data sources. The final version of the ontology consists in roughly 180 concepts and 170 roles and attributes, and contains about 1000 OWL 2 assertions.

GRAPHOL is also currently used in a joint project with ACI, the Italian Automobile Club, for modeling the domain of the car tax system, and it is used in projects with partners such as Consip and ISTAT. GRAPHOL was deployed at Telecom Italia at [http://www.dis.uniroma1.it/~ontodeb/](http://www.dis.uniroma1.it/~ontodeb/) (in Italian) using “guest” for both login and password.

---

6GRAPHOL diagrams of the ontology are accessible at [http://www.dis.uniroma1.it/~ontodeb/](http://www.dis.uniroma1.it/~ontodeb/).

7[http://www.telecomitalia.it/](http://www.telecomitalia.it/)


9[http://www.aci.it](http://www.aci.it)

10[http://www.consip.it/](http://www.consip.it/)

11[http://www.istat.it/](http://www.istat.it/)
also adopted internally for designing the ontology for one of the use-cases in Optique [76, 77], a large-scale integrating project (IP) of the FP7 framework.

Another example of the use of Graphol is the Planergy project [12] for energy monitoring in public administration buildings (the Graphol ontology produced during the project can be found at http://www.planergy.it/opendata/).

We finally point out that Graphol is also a valid teaching tool for introducing OWL and ontologies to people without logic background, as we successfully experienced in a recent advanced academic course on semantic technologies, mainly oriented to people from the industrial world, expert in information systems and information management.

5. User Evaluation Study

In this section, we discuss the setup and results of the user evaluation tests we have conducted for the Graphol language. The goal was to measure the effectiveness of Graphol for ontology understanding and design, both independently and when compared to other languages with similar purposes, by users with different backgrounds and varying levels of experience with ontologies and conceptual modeling.

We carried out two separate studies, designed around a series of model comprehension and model editing tasks.

5.1. Setup of the study

Before defining the definitive setup of the user study, we conducted two test runs with ontology and conceptual modeling experts who were already familiar with the Graphol language, which allowed us to iteratively improve the setup of the experiments. Our primary goals for these test runs was to verify that the tasks which we asked to perform were clear, and that their difficulty was adequate for the expertise of the final test participants. Examples of modifications that we made were the removal of over-complicated tasks, the modification of several questions to avoid ambiguity in what the questions were asking, the refinement of the cheat-sheets which were handed out with the questionnaires, and the definition of the time limits for each part of the experiments.

Because we conducted two different studies, with two groups of users and on different dates, we also took advantage of the experience of the first test to refine the second one. For instance, we made some modifications to the tutorial slides which were presented prior to the second test, in light of some doubts that were expressed by the users during the presentation of these slides during the first test.

5.2. Objectives of the study

The design of our user evaluation study was geared towards the achievement of two main objectives.

\[12\text{http://www.planergy.it/}\]
1. Evaluate the difficulty of using the GRAPHOL language for ontology comprehension and editing by users with advanced skills in conceptual modeling and (in some cases) basic skills in logics and ontologies.

2. Evaluate the difficulty of approaching and learning the GRAPHOL language for users with only basic skills in conceptual modeling and little or no experience with ontologies, both in isolation, and in comparison with another graphical ontology language that is strongly based on a formalism with which they are already familiar.

In accordance to these two objectives, we identified two groups of test participants, and defined two different types of tasks for each of these groups. Further details of the test participants, of the required tasks, and of the structure of the two tests are given in the remainder of this section. Here we give a brief sketch of these two tests: the first one included a series of comprehension tasks and of editing tasks on two ontology models represented in the GRAPHOL language; the second one, involving users with limited knowledge of conceptual modeling and ontologies, instead included two sets of comprehension tasks on two ontologies represented in GRAPHOL and in the OWLGrEd ontology language [42, 43].

The reason for limiting the comparative evaluation of GRAPHOL with OWLGrEd to the comprehension test with less skilled users is that the purpose of the test was to verify that the GRAPHOL language could be learned and used by these users with no more effort and difficulty than that needed for another language which is, by its very nature, more recognizable, due to its strong relation to UML class diagrams. Indeed, we are convinced that none of the graphical solutions for ontology representation discussed in Section 1 have succeeded in providing expert ontology users or designers with an effective tool to replace the more widely used non-graphical editors such as Protégé, and the fact that, to the best of our knowledge, these solutions have very rarely been adopted in real-life scenarios seems to corroborate this conviction.

Therefore, it was not in our interest to gauge the effectiveness of GRAPHOL against OWLGrEd, or any other similar visual language, among more expert users. Instead, our test with these users was specifically designed towards measuring their perception of GRAPHOL as a viable candidate for future use in the design of ontologies in real life.

Furthermore, practical considerations led us to avoid a comparative study between GRAPHOL and OWLGrEd for ontology editing purposes. Firstly, incorporating a comparison between these two languages in the tests would have drawn out their duration excessively (as it is, the tests exceeded two hours). Secondly, a comparison of the editing capabilities of these two languages would have been strongly influenced by the respective tools used for the task, and an evaluation of these tools was outside the scope of the experiments.

Finally, we also did not conduct a comparative evaluation of GRAPHOL against ontology editors such as Protégé, or simply against modeling through DL axioms, because these approaches obviously do not offer graphical editing solutions, whereas our main aim has been to test the evaluation of consumers and designers not necessary skilled in ontologies.
5.3. Participants

Participants were recruited among computer science researchers, post-docs, Ph.D. and master’s students. Eighteen participants took part in the test; ten with only basic skills in conceptual modeling and little or no experience with ontologies (beginners), and eight with advanced knowledge in conceptual modeling and basic skills in logic and ontology design (experts). Table 4 recaps some descriptive statistics about the users regarding age, highest completed education degree, years of experience with ontologies, and knowledge of conceptual modeling and ontologies. Note that, as expected, the knowledge of conceptual modeling among experts is in general very high (4.2 out of 5 average, with a low standard deviation of 0.7), and is fairly high among beginners (3.3 out of 5, again with a low standard deviation of 0.7). Furthermore, the average knowledge of ontologies is lower than that of conceptual modeling for both experts and beginners.

<table>
<thead>
<tr>
<th>Age</th>
<th>Education</th>
<th>Conceptual Modeling Knowledge</th>
<th>Ontology Knowledge</th>
</tr>
</thead>
<tbody>
<tr>
<td>min</td>
<td>22</td>
<td>27</td>
<td>1</td>
</tr>
<tr>
<td>max</td>
<td>28</td>
<td>47</td>
<td>2</td>
</tr>
<tr>
<td>median</td>
<td>24.5</td>
<td>31</td>
<td>1</td>
</tr>
<tr>
<td>mean</td>
<td>24.7</td>
<td>34.2</td>
<td>1.1</td>
</tr>
<tr>
<td>st.dev.</td>
<td>2.3</td>
<td>6.7</td>
<td>0.3</td>
</tr>
</tbody>
</table>

Table 4: Statistics of the participants (“Beg.” indicates statistics for beginners, “Exp.” indicates statistics for experts): for Education, 1 = Bachelor Degree, 2 = Master’s Degree, 3 = Ph.D; Conceptual Modeling and Ontology Knowledge are on a scale from 1 to 5, with 1 indicating no knowledge.

5.4. Ontology models

We chose three different ontologies for the study, the Pizza ontology, the Lehigh University Benchmark (LUBM) ontology, and the Family ontology, and modeled excerpts of the Pizza and LUBM ontologies in both GRAPHOL and OWLGrEd, and of the Family ontology in GRAPHOL. These ontologies were chosen for their popularity among the Semantic Web community, and due to the fact that the simple and widely-understood nature of the domain of these ontologies guarantees that the results of the test would not be altered by misinterpretation of the meanings of the terms in the ontologies.

The ontology models can be found in appendix to each of the tests, which we provide at the end of this document. We remark that in the GRAPHOL models we have sometimes used some graphical shortcuts that are defined in GRAPHOL, and that we have not mentioned in the paper for sake of brevity.

5.5. Language for comparison

Among the available language candidates for the comparative test, we chose OWLGrEd \[42, 43\] which provides a graphical notation for OWL 2 based on UML class diagrams. OWLGrEd was chosen because its goal and its expressive power are akin to Graphol’s, while its UML-based design principles and visual representation are quite different from Graphol, but are, at least in principle, easily accessible to users who are familiar with UML class diagrams. As stated earlier, among our goals was to evaluate the difficulty of approaching the Graphol language for a non-expert user, in comparison to that of learning a language based on a formalism with which the user is familiar with, and OWLGrEd is an ideal fit for this task.

The comparative aspect of the study was limited to these two languages, in order to avoid encumbering the users with an excessive amount of new information to process during the tests.

Here we give some further details about OWLGrEd, and in Figures 7 and 8 provide a very simple example of a model represented respectively in Graphol and OWLGrEd. For a complete presentation of OWLGrEd, we refer the reader to \[42\] and \[43\].

The OWLGrEd notation is based on UML class diagrams. Specifically:

- OWLGrEd concepts are represented as UML classes, without operations;
- OWLGrEd attributes are represented as UML class attributes, but with different default cardinalities;
- OWLGrEd roles are represented as UML binary associations with the arrow indicating the direction, from the domain to the range. OWLGrEd roles are thus typed in both the domain and the range;
- OWLGrEd cardinalities on roles are represented as UML cardinalities on roles, with the possibility of further refining the cardinality;
- OWLGrEd cardinality restrictions on attributes are represented as UML cardinalities on attributes;
- OWLGrEd inclusions between concepts are represented as UML ISAs between classes;
- OWLGrEd generalizations are represented as UML generalizations, using a special graphical symbol;
- OWLGrEd uses the OWL 2 Manchester syntax\[16\] to specify expressions which denote complex concepts;
- OWLGrEd role restrictions are represented as red arrows from the concept which is included in the restriction to the concept that qualifies the restriction, labeled with the name of the restricted role.

\[16\]http://www.w3.org/TR/owl2-manchester-syntax/
5.6. Tasks

We designed a series of model comprehension tasks for both the Pizza and LUBM ontology models, and of model modification tasks for the Family ontology model. The tasks were designed to present a varying degree of difficulty to the user: those for beginners were limited to more basic aspects of conceptual modeling, while those presented to experts also focused on slightly more advanced aspects.

Each comprehension task consisted in answering a question regarding the domain represented by the given model. Question types vary, from open format questions to closed format (or multiple choice) questions, to yes or no questions. Each modification task instead requested the user to modify the given GRAPHOL model of the Family ontology by modeling one or more assertions, provided in natural language. The complete set of tests is provided at the end of this document.

For each task, the participant was asked to measure the time in minutes in which he completed the task. We also asked each participant to indicate, on a scale from 0 to 4, the clarity and the easiness of each task. To understand the difference between these two response variables, consider that the participant may think he has clearly understood what must do for a certain task, but may not be able to easily put it into practice, or vice-versa. In other words, the first variable is a measure of the quality of the questionnaire, while the second of the tools the user is provided with to carry out the tasks. Examples of the two
question types are provided below.

- Was the question clear?
  Not clear at all  □  □  □  □  □  Very clear
- Were you able to easily answer this question?
  Not at all  □  □  □  □  □  Absolutely

5.7. Structure of the study

Here we provide the details for the evaluation studies of beginners and experts, which were conducted separately, on different dates.

**Beginners:**

1. *Introduction and brief Graphol tutorial (15 minutes):* a general introduction to the purpose of the experiment, and a brief tutorial on ontologies and on the Graphol language.

2. *Brief OWLGrEd tutorial (15 minutes):* a brief tutorial on the OWLGrEd language.

3. *Brief user background questionnaire (5 minutes):* participants had to answer a brief questionnaire in which they were asked to provide some personal background information, as well as to rate their knowledge of conceptual modeling and ontologies on a scale from 1 to 5 (with 1 indicating extremely low and 5 extremely high expertise), to indicate how many years they had of experience with ontologies (if any), whether they were familiar with some of the more popular ontology editors and knowledge representation and conceptual modeling formalisms, and whether they had any experience with ontologies in real-life scenarios or in manually creating or editing ontologies.

4. *LUBM comprehension tasks (40 minutes):* each user was asked to answer ten questions on the LUBM model they were provided. Half of the users were provided a Graphol version of the LUBM model, and half an OWLGrEd version.

5. *Pizza comprehension tasks (40 minutes):* each user was asked to answer ten questions on the Pizza model they were provided. Half of the users (those which were provided the OWLGrEd version of the LUBM model) were provided a Graphol version of the Pizza model, and half an OWLGrEd version.

6. *Ex-post survey (10 minutes):* after all the comprehension tasks were completed, the users were asked to fill in a brief survey regarding their experience. The survey required the users to rate, on a scale from 0 to 4, the general difficulty of the comprehension tasks, the difficulty of learning the Graphol and OWLGrEd symbols, the difficulty of using Graphol and OWLGrEd to read ontologies, and, optionally, to indicate aspects of Graphol and OWLGrEd which they particularly liked, or that they would like to see improved.

**Experts:**

26
1. **Introduction and brief Graphol tutorial (30 minutes):** the participants were given the same introductory tutorial on ontologies and on the Graphol language as the beginners, with the addition of some more complex features on the Graphol language which were featured in the expert questionnaire but not the beginner questionnaire.

2. **Brief user background questionnaire (5 minutes):** the participants had to answer the same background questionnaire given to the beginners.

3. **Graphol comprehension tasks (35 minutes):** after completing the introductory part on Graphol, each user was asked to answer ten questions on the Graphol model of the Pizza ontology they were provided.

4. **Graphol editing tasks (35 minutes):** each user was asked to carry out ten editing tasks on the Graphol model of the Family ontology they were provided.

5. **Ex-post survey (5 minutes):** after carrying out both the comprehension and editing tasks, the users were asked to fill out a brief survey, analogous to the one given to the beginners.

We now discuss some more detailed aspects of the study.

- All participants, in support of their tasks, were provided with documentation regarding the language in play for that specific task. Specifically, the questionnaire included some cheat sheets which recapped the symbols of the Graphol language and their meaning, along with some examples of the Graphol representation of some of the most common ontology expressions and assertions. Additionally, users were provided with a printout of the slides of the introductory tutorial on Graphol.

- The order in which the tasks were presented in the questionnaires was intentionally random, i.e., not linked to the expected difficulty of each task. This choice was made in order to compensate for a potential bias given by the learning curve of familiarizing with Graphol or OWLGrEd during the course of the tasks. In other words, we wanted to avoid facilitating the participants by allowing them to face easier questions at the beginning of each task, and more difficult ones at the end, when they would probably have gained familiarity with the language.

- The experimental design method we chose for the comparative study between Graphol and OWLGrEd is the within-subjects method, common in HCI [80]. This choice, as opposed to the between-subjects technique, was made mainly due to the limited number of participants to the experiment. So, each user was asked to complete the comprehension tasks both for the Graphol language and for the OWLGrEd language. In order to avoid the transfer of learning effects between tasks, we split the ten users in two groups of five, and asked the first group to first carry out the comprehension tasks on the Graphol version of the LUBM model, and then the comprehension tasks on the OWLGrEd version of the Pizza model, and the second group to do the opposite.

The questionnaires can be found in the Appendix of this paper.
5.8. Study results

Figure 9 summarizes some of the results of the tests. Each box plot shows the full range of variation, from minimum to maximum, indicated by the whiskers, the likely range of variation, indicated by the two boxes, and the median value. The overall correctness results for the comparative test, shown in boxplot (a), indicate a good level of comprehension by the novice users of the Graphol language, comparable to the one showed for OWLGrEd, which is strongly based on a formalism (UML) which they were familiar with. Boxplot (b) summarizes the correctness results of the comparative test limited to five questions that deal with complex modeling aspects that go beyond UML, i.e., questions 1,3,5,9, and 10 on the LUBM questionnaires, and questions 2,4,5,7, and 9 on the Pizza questionnaires. These results confirm that users were able to more easily understand the completely graphical representation provided by Graphol than the formulas or non-UML constructs adopted in OWLGrEd. Finally, boxplot (c) shows the correctness results of the non-comparative test by the more expert users. The very high scores indicate that such users quickly learned how to read Graphol diagrams and how to use the language for modeling.

5.9. Post-questionnaire analysis

Finally, we discuss the results of the post-test questionnaires, which were presented to all participants. We recall that the goal of the questionnaires was to measure the perceived general difficulty of the tasks, of learning the symbols of the languages used in the test, and of using Graphol to read ontologies. The average results of the questionnaires are shown in Figure 10.

As one can see, the feedback essentially confirms the positive impression gained from the analysis of the test results. Users felt about as comfortable reading ontologies modeled through the novel Graphol language as they were with those in OWLGrEd (almost identical average scores for “Reading Difficulty”), even though, as expected, they felt that learning Graphol’s symbols was slightly more difficult than learning OWLGrEd’s UML-based ones (“Symbol Difficulty” scores). The average values in the “Task Difficulty” column are more a reflection of the difficulty of the tests rather than the languages, but in

---

17More details on our user studies, including the tests, can be found in the supplemental material attached to this submission.
both cases users seem to feel that the tests were not excessively difficult (average difficulty score of 2.4 and 2.2 out of a maximum of 5).

6. Conclusion and Future Works

In this paper we have presented GRAPHOL, a novel language for the visual representation of ontologies. We have shown that GRAPHOL is based on a standard language for conceptual modeling, highlighting that its core components are based on the Entity-Relationship model, and that its key feature is that it allows a completely graphical specification of ontologies, with no need for logical formulae in the diagram. This characteristic makes it particularly suitable even for users who do not have a strong background in “classical” logic-based ontology languages. Furthermore, GRAPHOL has a precise syntax and semantics through an encoding in DLs which allows it to encompass the highly expressive $SROIQ(D)$ DL. We have also given some examples of industrial projects in which GRAPHOL is currently used, underlining that GRAPHOL has played a key part in the success of such projects. Finally, we have carried out and described some user evaluation studies which show the effectiveness of the usage of GRAPHOL as a diagrammatic language for OWL2.

We are currently planning to add additional features to the language itself, to allow an even broader scope of modelling possibilities. On one hand, we are interested in extending GRAPHOL by including n-ary relations, which would enhance its compatibility with ER. On the other, we are looking into the addition of metamodeling features such as metaconcepts, which are concepts whose instances can be concepts themselves, and metaproperties, which are relationships between metaconcepts. Metaconcept representation could be useful, for instance, for the representation in GRAPHOL of formal ontologies, where specific metaproperties, such as rigidity, are used to express relevant aspects of the intended meaning of the elements in an ontology.

References


[31] C. M. Keet, Mapping the Object-Role Modeling language ORM2 into description logic language $\mathcal{DLR}_{idf}$, arXiv preprint cs/0702089.


G. A. Miller, The magical number seven, plus or minus two: some limits on our capacity for processing information., Psychological review 63 (2) (1956) 81.


[77] M. Giese, D. Calvanese, P. Haase, I. Horrocks, Y. Ioannidis, H. Killapi, M. Koubarakis, M. Lenzerini, R. M’oller, M. Rodriguez-Muro, O. ’Ozcep,


Graphol: a graphical ontology language
Survey and User Evaluation Study

Purpose of the Questionnaire
Evaluate the usability of the Graphol language for ontology modeling through a series of comprehension tasks.

Pre-questionnaire
We would like for you to provide some personal and professional background information, regarding experience with ontology modelling, knowledge representation, and ontology languages (OWL2, Description Logics). Please fill in the following short survey.

1. Sex □ Male □ Female
2. How old are you? ______
3. Please describe your occupation/profession and role?

4. What is type and field of your highest completed education degree?

5. Roughly how many years of experience with ontologies do you have (0 if none)? ______

6. On a scale from 1 to 5, how would you rate your knowledge of conceptual modeling?
   None □ □ □ □ □ Very high

7. On a scale from 1 to 5, how would you rate your knowledge of ontologies?
   None □ □ □ □ □ Very high

8. Which ontology editors are you familiar with?
   □ Protégé □ TopBraid Composer □ NeOn Toolkit □ OWLGrEd
   □ OntoStudio □ OntoUML □ Other:

9. Which knowledge representation and conceptual modeling formalisms are you familiar with?
   □ OWL2 □ E-R □ Description Logics □ UML Class Diagrams
   □ ORM □ Other:

10. Have you ever had experience in manually creating or editing an ontology (Y/N)? ______

11. Have you ever had experience in working with medium- or large-scale ontologies in real-life projects (Y/N)? ______

12. Which ontology visualization tools are you familiar with?
   □ OntoGraf □ OWLViz □ GROWL □ OWLGrEd □ Other:
Graphol Cheat Sheet - 1

Graphical symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
<th>Symbol</th>
<th>Meaning</th>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Concept</td>
<td></td>
<td>Intersection</td>
<td></td>
<td>Domain</td>
</tr>
<tr>
<td></td>
<td>Role</td>
<td></td>
<td>Union</td>
<td></td>
<td>Range</td>
</tr>
<tr>
<td></td>
<td>Attribute</td>
<td></td>
<td>Complement</td>
<td></td>
<td>Inclusion edge</td>
</tr>
<tr>
<td></td>
<td>Value domain</td>
<td></td>
<td>Inverse</td>
<td></td>
<td>Parameter</td>
</tr>
<tr>
<td></td>
<td>Disjoint Union</td>
<td></td>
<td></td>
<td></td>
<td>input edge</td>
</tr>
</tbody>
</table>

Restriction types:
- existential: “exists”
- universal: “forall”
- cardinality: “x,y”, with y ≥ x, and x = “-“ if no minimum cardinality is specified, and y = “-“ if no maximum cardinality is specified

If the restriction type label on a white or black square is missing, then the existential restriction is implied.
### Intentional assertions

In the following table, the “Concept” and “Role” nodes in the Graphol column can be replaced with complex concepts and roles.

<table>
<thead>
<tr>
<th>Meaning</th>
<th>Graphol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concept inclusion</td>
<td><img src="image1" alt="Concept inclusion" /></td>
</tr>
<tr>
<td>Role inclusion</td>
<td><img src="image2" alt="Role inclusion" /></td>
</tr>
<tr>
<td>Attribute inclusion</td>
<td><img src="image3" alt="Attribute inclusion" /></td>
</tr>
<tr>
<td>Concept equivalence</td>
<td><img src="image4" alt="Concept equivalence" /></td>
</tr>
<tr>
<td>Role equivalence</td>
<td><img src="image5" alt="Role equivalence" /></td>
</tr>
<tr>
<td>Attribute equivalence</td>
<td><img src="image6" alt="Attribute equivalence" /></td>
</tr>
<tr>
<td>Globally functional role</td>
<td><img src="image7" alt="Globally functional role" /></td>
</tr>
<tr>
<td>Globally inverse functional role</td>
<td><img src="image8" alt="Globally inverse functional role" /></td>
</tr>
<tr>
<td>Globally functional and inverse functional role</td>
<td><img src="image9" alt="Globally functional and inverse functional role" /></td>
</tr>
<tr>
<td>Globally functional attribute</td>
<td><img src="image10" alt="Globally functional attribute" /></td>
</tr>
</tbody>
</table>
**Graphol Cheat Sheet - 3**

In the following table we provide the most common expressions. Graphol n-ary operators with \( n > 2 \) are represented as having 3 input parameters. Restrictions for Graphol role domains exist also for role ranges. The “Concept” and “Role” nodes in the Graphol column can be replaced with complex concepts and roles.

<table>
<thead>
<tr>
<th>Meaning</th>
<th>Graphol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concept</td>
<td></td>
</tr>
<tr>
<td>Concept intersection</td>
<td>![Diagram of Concept Intersection]</td>
</tr>
<tr>
<td>Concept union</td>
<td>![Diagram of Concept Union]</td>
</tr>
<tr>
<td>Concept complement</td>
<td>![Diagram of Concept Complement]</td>
</tr>
<tr>
<td>Universal role domain restriction</td>
<td>![Diagram of Universal Role Domain Restriction]</td>
</tr>
<tr>
<td>Universal attribute domain restriction</td>
<td>![Diagram of Universal Attribute Domain Restriction]</td>
</tr>
<tr>
<td>Existential role domain restriction</td>
<td>![Diagram of Existential Role Domain Restriction]</td>
</tr>
<tr>
<td>Existential attribute domain restriction</td>
<td>![Diagram of Existential Attribute Domain Restriction]</td>
</tr>
<tr>
<td>Min. and max. cardinality role restriction</td>
<td>![Diagram of Min. and Max. Cardinality Role Restriction]</td>
</tr>
<tr>
<td>Min. and max. cardinality attribute restriction</td>
<td>![Diagram of Min. and Max. Cardinality Attribute Restriction]</td>
</tr>
<tr>
<td>Role</td>
<td>![Diagram of Role]</td>
</tr>
<tr>
<td>Inverse role</td>
<td>![Diagram of Inverse Role]</td>
</tr>
<tr>
<td>Attribute</td>
<td>![Diagram of Attribute]</td>
</tr>
</tbody>
</table>
Graphol Cheat Sheet - 4

In the following table you can see some of the more typical examples of conceptual design in Graphol. The “Concept” and “Role” nodes in the Graphol column can be replaced with complex concepts and roles.

<table>
<thead>
<tr>
<th>Relation</th>
<th>Graphol</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISA between classes</td>
<td><img src="image1" alt="ISA between classes" /></td>
</tr>
<tr>
<td>Disjointness between classes</td>
<td><img src="image2" alt="Disjointness between classes" /></td>
</tr>
<tr>
<td>ISA between roles</td>
<td><img src="image3" alt="ISA between roles" /></td>
</tr>
<tr>
<td>Disjointness between roles</td>
<td><img src="image4" alt="Disjointness between roles" /></td>
</tr>
<tr>
<td>Typings of the domain and range of a role</td>
<td><img src="image5" alt="Typings of the domain and range of a role" /></td>
</tr>
<tr>
<td>Typings of the domain and range of an attribute</td>
<td><img src="image6" alt="Typings of the domain and range of an attribute" /></td>
</tr>
<tr>
<td>Cardinality restriction on the mandatory participation of concepts to roles/attributes</td>
<td><img src="image7" alt="Cardinality restriction on the mandatory participation of concepts to roles/attributes" /></td>
</tr>
<tr>
<td>Global functionality assertions</td>
<td><img src="image8" alt="Global functionality assertions" /></td>
</tr>
</tbody>
</table>
**Comprehension Task**

You have been provided a printout of a simple Graphol model which depicts an extract of the Lehigh University Benchmark (LUBM) ontology (also see Appendix A). The model represents a university domain ontology.

Please answer the following questions.

1. Must a GraduateStudent attend only GraduateLevelCourses(Y/N)?

   Start time: ___________  Finishing time: ___________

   Was the question clear? Not clear at all □□□□□ Very clear
   Were you able to easily answer this question? Not at all □□□□□ Absolutely

2. List the pairs of atomic sub-concepts of Employee in which the two concepts are disjoint from each other.

   ______________________________________________________

   Start time: ___________  Finishing time: ___________

   Was the question clear? Not clear at all □□□□□ Very clear
   Were you able to easily answer this question? Not at all □□□□□ Absolutely

3. Can a Research Assistant work for a School(Y/N)?

   Start time: ___________  Finishing time: ___________

   Was the question clear? Not clear at all □□□□□ Very clear
   Were you able to easily answer this question? Not at all □□□□□ Absolutely

4. Can Student and Employee have a common instance(Y/N)?

   Start time: ___________  Finishing time: ___________

   Was the question clear? Not clear at all □□□□□ Very clear
   Were you able to easily answer this question? Not at all □□□□□ Absolutely

5. For each of the following roles and attributes, state whether the given concept can participate in it, and, if so, specify if the participation is mandatory.

<table>
<thead>
<tr>
<th>Role/Attribute</th>
<th>Yes</th>
<th>No</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>a) Employee - worksFor:</td>
<td>□ Yes</td>
<td>□ No</td>
<td>□ Mandatory</td>
</tr>
<tr>
<td>b) Student - attends:</td>
<td>□ Yes</td>
<td>□ No</td>
<td>□ Mandatory</td>
</tr>
<tr>
<td>c) Person - telephoneNumber:</td>
<td>□ Yes</td>
<td>□ No</td>
<td>□ Mandatory</td>
</tr>
<tr>
<td>d) University - hasAlumnus:</td>
<td>□ Yes</td>
<td>□ No</td>
<td>□ Mandatory</td>
</tr>
<tr>
<td>e) Person - title:</td>
<td>□ Yes</td>
<td>□ No</td>
<td>□ Mandatory</td>
</tr>
<tr>
<td>f) PostDoctorate - teaches:</td>
<td>□ Yes</td>
<td>□ No</td>
<td>□ Mandatory</td>
</tr>
</tbody>
</table>

   Start time: ___________  Finishing time: ___________
   Was the question clear? Not clear at all □□□□□ Very clear
   Were you able to easily answer this question? Not at all □□□□□ Absolutely

6. Specify whether the following roles and attributes are functional, inverse functional, or both.

<table>
<thead>
<tr>
<th>Role/Attribute</th>
<th>Functional</th>
<th>Inverse functional</th>
</tr>
</thead>
<tbody>
<tr>
<td>a) title:</td>
<td>□</td>
<td></td>
</tr>
<tr>
<td>b) worksFor:</td>
<td>□</td>
<td></td>
</tr>
<tr>
<td>c) teaches:</td>
<td>□</td>
<td></td>
</tr>
<tr>
<td>d) email:</td>
<td>□</td>
<td></td>
</tr>
</tbody>
</table>

   Start time: ___________  Finishing time: ___________
   Was the question clear? Not clear at all □□□□□ Very clear
   Were you able to easily answer this question? Not at all □□□□□ Absolutely
7. List the typings of the domain and range of the following attributes.

a) title
   Domain: ___________________________ Range: ___________________________

b) name
   Domain: ___________________________ Range: ___________________________

c) age
   Domain: ___________________________ Range: ___________________________

8. List the typings of the domain and range of the following roles.

a) hasAlumnus
   Domain: ___________________________ Range: ___________________________

b) publishes
   Domain: ___________________________ Range: ___________________________

c) hasDegreeFrom
   Domain: ___________________________ Range: ___________________________

d) worksFor
   Domain: ___________________________ Range: ___________________________

9. Name two roles that are one the inverse of the other.

10. Must each Person have a telephone number(Y/N)? _____
Post-test questionnaire

Please answer the following questions at the end of your test.

1. How would you rate the general difficulty of the comprehension tasks of the test?
   Very easy  □□□□□ Very difficult

2. How would you rate the difficulty of learning the symbols of the Graphol language?
   Very easy  □□□□□ Very difficult

3. How would you rate the general difficulty of using Graphol for reading ontologies?
   Very easy  □□□□□ Very difficult

4. (Optional) Aspects of the Graphol language that you particularly like:
   __________________________________________________________
   __________________________________________________________
   __________________________________________________________
   __________________________________________________________
   __________________________________________________________
   __________________________________________________________
   __________________________________________________________
   __________________________________________________________

5. (Optional) Aspects of the Graphol language that you would like to see improved/changed:
   __________________________________________________________
   __________________________________________________________
   __________________________________________________________
   __________________________________________________________
   __________________________________________________________
   __________________________________________________________
   __________________________________________________________
   __________________________________________________________
Appendix A

Model used for Graphol Comprehension Task
OWLGrEd: a graphical ontology language
Survey and User Evaluation Study

Purpose of the Questionnaire
Evaluate the usability of the OWLGrEd language for ontology modeling through a series of comprehension tasks.

OWLGrEd Cheat Sheet - 1

Graphical symbols
The “Concept”, “Role”, “Attribute”, and “Value-domain” labels on the symbols indicate the name of the respective ontology predicate.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
<th>Symbol</th>
<th>Meaning</th>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>![Concept]</td>
<td>Concept</td>
<td>![Disjoint hierarchy connector]</td>
<td>Disjoint hierarchy connector</td>
<td>![Restriction edge]</td>
<td>Restriction edge</td>
</tr>
<tr>
<td>![Role]</td>
<td>Role</td>
<td>![Complete hierarchy connector]</td>
<td>Complete hierarchy connector</td>
<td>![Equivalence edge]</td>
<td>Equivalence edge</td>
</tr>
<tr>
<td>![Attribute]</td>
<td>Attribute</td>
<td>![Inclusion edge]</td>
<td>Inclusion edge</td>
<td>![Disjunction edge]</td>
<td>Disjunction edge</td>
</tr>
<tr>
<td>![Value domain]</td>
<td>Value domain</td>
<td>![Input edge]</td>
<td>Input edge</td>
<td>![Complement edge]</td>
<td>Complement edge</td>
</tr>
<tr>
<td>![Hierarchy connector]</td>
<td>Hierarchy connector</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**OWLGrEd Cheat Sheet - 2**

In the following table you can see some of the more typical examples of conceptual design in OWLGrEd. The “Concept” label on concept nodes in the OWLGrEd column can be replaced with a complex expression label, which indicates that the concept node represents a concept expression such as the union or intersection of concepts. In OWLGrEd, concept inclusion can alternatively be represented through an inclusion label, “<”, followed by the including concept, in the concept node.

<table>
<thead>
<tr>
<th>Relation</th>
<th>Graphol</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISA between concepts</td>
<td>![Graph for ISA between concepts]</td>
</tr>
<tr>
<td>Disjointness between concepts</td>
<td>![Graph for Disjointness between concepts]</td>
</tr>
<tr>
<td>Equivalence between concepts</td>
<td>![Graph for Equivalence between concepts]</td>
</tr>
<tr>
<td>ISA between roles</td>
<td>![Graph for ISA between roles]</td>
</tr>
<tr>
<td>Disjointness between roles</td>
<td>![Graph for Disjointness between roles]</td>
</tr>
<tr>
<td>Equivalence between roles</td>
<td>![Graph for Equivalence between roles]</td>
</tr>
<tr>
<td>ISA between attributes</td>
<td>![Graph for ISA between attributes]</td>
</tr>
<tr>
<td>Disjointness between attributes</td>
<td>![Graph for Disjointness between attributes]</td>
</tr>
<tr>
<td>Equivalence between attributes</td>
<td>![Graph for Equivalence between attributes]</td>
</tr>
<tr>
<td>Typings of the domain and range of a role</td>
<td>![Graph for Typings of the domain and range of a role]</td>
</tr>
<tr>
<td>Typings of the domain and range of an attribute</td>
<td>![Graph for Typings of the domain and range of an attribute]</td>
</tr>
<tr>
<td>Cardinality of roles/attributes</td>
<td>![Graph for Cardinality of roles/attributes]</td>
</tr>
<tr>
<td>Restriction on roles/attributes</td>
<td>![Graph for Restriction on roles/attributes]</td>
</tr>
</tbody>
</table>

Restriction-type = some/only/exactly x/max x/x/min x

| Functionality assertions                      | ![Graph for Functionality assertions] |

Restriction-type = some/only/exactly x/max x/x/min x
Comprehension Task

You have been provided a printout of a simple OWLGrEd model which depicts an extract of the Lehigh University Benchmark (LUBM) ontology (also see Appendix A). The model represents a university domain ontology.

Please answer the following questions.

1. Must a GraduateStudent attend only GraduateLevelCourses(Y/N)?

Start time: ___________  Finishing time: ___________

Was the question clear? Not clear at all □—□—□—□—□ Very clear
Were you able to easily answer this question? Not at all □—□—□—□—□ Absolutely

2. List the pairs of atomic sub-concepts of Employee in which the two concepts are disjoint from each other.

Start time: ___________  Finishing time: ___________

Was the question clear? Not clear at all □—□—□—□—□ Very clear
Were you able to easily answer this question? Not at all □—□—□—□—□ Absolutely

3. Can a Research Assistant work for a School(Y/N)?

Start time: ___________  Finishing time: ___________

Was the question clear? Not clear at all □—□—□—□—□ Very clear
Were you able to easily answer this question? Not at all □—□—□—□—□ Absolutely

4. Can Student and Employee have a common instance(Y/N)?

Start time: ___________  Finishing time: ___________

Was the question clear? Not clear at all □—□—□—□—□ Very clear
Were you able to easily answer this question? Not at all □—□—□—□—□ Absolutely

5. For each of the following roles and attributes, state whether the given concept can participate in it, and, if so, specify if the participation is mandatory.

   a) Employee - worksFor: □ Yes □ No □ Mandatory
   b) Student - attends: □ Yes □ No □ Mandatory
   c) Person - telephoneNumber: □ Yes □ No □ Mandatory
   d) University - hasAlumnus: □ Yes □ No □ Mandatory
   e) Person - title: □ Yes □ No □ Mandatory
   f) PostDoctorate - teaches: □ Yes □ No □ Mandatory

Start time: ___________  Finishing time: ___________

Was the question clear? Not clear at all □—□—□—□—□ Very clear
Were you able to easily answer this question? Not at all □—□—□—□—□ Absolutely

6. Specify whether the following roles and attributes are functional, inverse functional, or both.

   a) title: □ Functional □ Inverse functional
   b) worksFor: □ Functional □ Inverse functional
   c) teaches: □ Functional □ Inverse functional
   d) email: □ Functional □ Inverse functional
Start time: _____________  Finishing time: _____________
Was the question clear? Not clear at all ⊗—⊗—⊗—⊗ Very clear
Were you able to easily answer this question? Not at all ⊗—⊗—⊗—⊗ Absolutely

7. List the typings of the domain and range of the following attributes.
   a) title  Domain: ____________________________  Range: ____________________________
   b) name  Domain: ____________________________  Range: ____________________________
   c) age  Domain: ____________________________  Range: ____________________________

Start time: _____________  Finishing time: _____________
Was the question clear? Not clear at all ⊗—⊗—⊗—⊗ Very clear
Were you able to easily answer this question? Not at all ⊗—⊗—⊗—⊗ Absolutely

8. List the typings of the domain and range of the following roles.
   a) hasAlumnus  Domain: ____________________________  Range: ____________________________
   b) publishes  Domain: ____________________________  Range: ____________________________
   c) hasDegreeFrom  Domain: ____________________________  Range: ____________________________
   d) worksFor  Domain: ____________________________  Range: ____________________________

Start time: _____________  Finishing time: _____________
Was the question clear? Not clear at all ⊗—⊗—⊗—⊗ Very clear
Were you able to easily answer this question? Not at all ⊗—⊗—⊗—⊗ Absolutely

9. Name two roles that are one the inverse of the other.

Start time: _____________  Finishing time: _____________
Was the question clear? Not clear at all ⊗—⊗—⊗—⊗ Very clear
Were you able to easily answer this question? Not at all ⊗—⊗—⊗—⊗ Absolutely

10. Must each Person have a telephone number (Y/N)? _____
Start time: _____________  Finishing time: _____________
Was the question clear? Not clear at all ⊗—⊗—⊗—⊗ Very clear
Were you able to easily answer this question? Not at all ⊗—⊗—⊗—⊗ Absolutely
Post-test questionnaire

Please answer the following questions at the end of your test.

1. **How would you rate the general difficulty of the comprehension tasks of the test?**
   Very easy □□□□□ Very difficult

2. **How would you rate the difficulty of learning the symbols of the OWLGrEd language?**
   Very easy □□□□□ Very difficult

3. **How would you rate the general difficulty of using OWLGrEd for reading ontologies?**
   Very easy □□□□□ Very difficult

4. (Optional) Aspects of the OWLGrEd language that you particularly like:

   __________________________________________________________
   __________________________________________________________
   __________________________________________________________
   __________________________________________________________
   __________________________________________________________
   __________________________________________________________

5. (Optional) Aspects of the OWLGrEd language that you would like to see improved/changed:

   __________________________________________________________
   __________________________________________________________
   __________________________________________________________
   __________________________________________________________
   __________________________________________________________
   __________________________________________________________
Appendix A

Model used for Comprehension Task
Graphol: a graphical ontology language
Survey and User Evaluation Study

Purpose of the Questionnaire
Evaluate the usability of the Graphol language for ontology modeling through a series of comprehension tasks.

Pre-questionnaire
We would like for you to provide some personal and professional background information, regarding experience with ontology modelling, knowledge representation, and ontology languages (OWL2, Description Logics). Please fill in the following short survey.

1. Sex  □ Male  □ Female

2. How old are you? ______

3. Please describe your occupation/profession and role?

4. What is type and field of your highest completed education degree?

5. Roughly how many years of experience with ontologies do you have (0 if none)? ______

6. On a scale from 1 to 5, how would you rate your knowledge of conceptual modeling?
   None □□□□□— Very high

7. On a scale from 1 to 5, how would you rate your knowledge of ontologies?
   None □□□□□— Very high

8. Which ontology editors are you familiar with?
   □ Protégé  □ TopBraid Composer  □ NeOn Toolkit  □ OWLGrEd
   □ OntoStudio  □ OntoUML  □ Other:

9. Which knowledge representation and conceptual modeling formalisms are you familiar with?
   □ OWL2  □ E-R  □ Description Logics  □ UML Class Diagrams
   □ ORM  □ Other:

10. Have you ever had experience in manually creating or editing an ontology (Y/N)? ______

11. Have you ever had experience in working with medium- or large-scale ontologies in real-life projects (Y/N)? ______

12. Which ontology visualization tools are you familiar with?
   □ OntoGraf  □ OWLViz  □ GROWL  □ OWLGrEd  □ Other:
Graphol Cheat Sheet - 1

Graphical symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
<th>Symbol</th>
<th>Meaning</th>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Concept</td>
<td></td>
<td>Intersection</td>
<td></td>
<td>Domain restriction</td>
</tr>
<tr>
<td></td>
<td>Role</td>
<td></td>
<td>Union</td>
<td></td>
<td>Range restriction</td>
</tr>
<tr>
<td></td>
<td>Attribute</td>
<td></td>
<td>Complement</td>
<td></td>
<td>Inclusion edge</td>
</tr>
<tr>
<td></td>
<td>Value domain</td>
<td></td>
<td>Inverse</td>
<td></td>
<td>Parameter input edge</td>
</tr>
<tr>
<td></td>
<td>Disjoint Union</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Restriction types:

- existential: “exists”
- universal: “forall”
- cardinality: “x,y”, with y ≥ x, and x = “-” if no minimum cardinality is specified, and y = “-” if no maximum cardinality is specified

If the restriction type label on a white or black square is missing, then the existential restriction is implied.
Graphol Cheat Sheet - 2

**Intentional assertions**

In the following table, the “Concept” and “Role” nodes in the Graphol column can be replaced with complex concepts and roles.

<table>
<thead>
<tr>
<th>Meaning</th>
<th>Graphol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concept inclusion</td>
<td><img src="image" alt="Concept inclusion Diagram" /></td>
</tr>
<tr>
<td>Role inclusion</td>
<td><img src="image" alt="Role inclusion Diagram" /></td>
</tr>
<tr>
<td>Attribute inclusion</td>
<td><img src="image" alt="Attribute inclusion Diagram" /></td>
</tr>
<tr>
<td>Concept equivalence</td>
<td><img src="image" alt="Concept equivalence Diagram" /></td>
</tr>
<tr>
<td>Role equivalence</td>
<td><img src="image" alt="Role equivalence Diagram" /></td>
</tr>
<tr>
<td>Attribute equivalence</td>
<td><img src="image" alt="Attribute equivalence Diagram" /></td>
</tr>
<tr>
<td>Globally functional role</td>
<td><img src="image" alt="Globally functional role Diagram" /></td>
</tr>
<tr>
<td>Globally inverse functional role</td>
<td><img src="image" alt="Globally inverse functional role Diagram" /></td>
</tr>
<tr>
<td>Globally functional and inverse functional role</td>
<td><img src="image" alt="Globally functional and inverse functional role Diagram" /></td>
</tr>
<tr>
<td>Globally functional attribute</td>
<td><img src="image" alt="Globally functional attribute Diagram" /></td>
</tr>
</tbody>
</table>
Graphol Cheat Sheet - 3

In the following table we provide the most common expressions. Graphol n-ary operators with \( n > 2 \) are represented as having 3 input parameters. Restrictions for Graphol role domains exist also for role ranges. The “Concept” and “Role” nodes in the Graphol column can be replaced with complex concepts and roles.

<table>
<thead>
<tr>
<th>Meaning</th>
<th>Graphol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concept</td>
<td><img src="image1" alt="Concept" /></td>
</tr>
<tr>
<td>Concept intersection</td>
<td><img src="image2" alt="Concept Intersection" /></td>
</tr>
<tr>
<td>Concept union</td>
<td><img src="image3" alt="Concept Union" /></td>
</tr>
<tr>
<td>Concept complement</td>
<td><img src="image4" alt="Concept Complement" /></td>
</tr>
<tr>
<td>Universal role domain restriction</td>
<td><img src="image5" alt="Universal Role Domain Restriction" /></td>
</tr>
<tr>
<td>Universal attribute domain restriction</td>
<td><img src="image6" alt="Universal Attribute Domain Restriction" /></td>
</tr>
<tr>
<td>Existential role domain restriction</td>
<td><img src="image7" alt="Existential Role Domain Restriction" /></td>
</tr>
<tr>
<td>Existential attribute domain restriction</td>
<td><img src="image8" alt="Existential Attribute Domain Restriction" /></td>
</tr>
<tr>
<td>Min. and max. cardinality role restriction</td>
<td><img src="image9" alt="Min. and Max. Cardinality Role Restriction" /></td>
</tr>
<tr>
<td>Min. and max. cardinality attribute restriction</td>
<td><img src="image10" alt="Min. and Max. Cardinality Attribute Restriction" /></td>
</tr>
<tr>
<td>Role</td>
<td><img src="image11" alt="Role" /></td>
</tr>
<tr>
<td>Inverse role</td>
<td><img src="image12" alt="Inverse Role" /></td>
</tr>
<tr>
<td>Attribute</td>
<td><img src="image13" alt="Attribute" /></td>
</tr>
</tbody>
</table>
Graphol Cheat Sheet - 4

In the following table you can see some of the more typical examples of conceptual design in Graphol. The “Concept” and “Role” nodes in the Graphol column can be replaced with complex concepts and roles.

<table>
<thead>
<tr>
<th>Relation</th>
<th>Graphol</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISA between classes</td>
<td><img src="image" alt="ISA between classes" /></td>
</tr>
<tr>
<td>Disjointness between classes</td>
<td><img src="image" alt="Disjointness between classes" /></td>
</tr>
<tr>
<td>ISA between roles</td>
<td><img src="image" alt="ISA between roles" /></td>
</tr>
<tr>
<td>Disjointness between roles</td>
<td><img src="image" alt="Disjointness between roles" /></td>
</tr>
<tr>
<td>Typings of the domain and range of a role</td>
<td><img src="image" alt="Typings of the domain and range of a role" /></td>
</tr>
<tr>
<td>Typings of the domain and range of an attribute</td>
<td><img src="image" alt="Typings of the domain and range of an attribute" /></td>
</tr>
<tr>
<td>Cardinality restriction on the mandatory participation of concepts to roles/attributes</td>
<td><img src="image" alt="Cardinality restriction on the mandatory participation of concepts to roles/attributes" /></td>
</tr>
<tr>
<td>Global functionality assertions</td>
<td><img src="image" alt="Global functionality assertions" /></td>
</tr>
</tbody>
</table>
Comprehension Task

You have been provided a printout of a simple Graphol model which depicts an extract of the popular Pizza ontology (also see Appendix A). The model essentially represents the fact that a pizza is composed of a base and one or more toppings, and shows how different toppings are matched with different kinds of pizza.

Please answer the following questions.

1. List the typings of the domain and range of the following roles.
   a) hasIngredient Domain: ______________________ Range: ______________________
   b) hasTopping Domain: ______________________ Range: ______________________
   c) hasBase Domain: ______________________ Range: ______________________

   Start time: ___________  Finishing time: ___________

   Were you able to easily answer this question? Not at all □—□—□—□ Absolutely □—□—□—□ Very clear

2. Must a Napoletana Pizza have a SpicyTopping(Y/N)? _____

   Start time: ___________  Finishing time: ___________

   Were you able to easily answer this question? Not at all □—□—□—□ Absolutely □—□—□—□ Very clear

3. Specify whether the following roles and attributes are functional, inverse functional, or both.
   a) hasIngredient: ☐ Functional ☐ Inverse functional
   b) hasTopping: ☐ Functional ☐ Inverse functional
   c) hasBase: ☐ Functional ☐ Inverse functional
   d) calories: ☐ Functional ☐ Inverse functional
   e) depth: ☐ Functional ☐ Inverse functional

   Start time: ___________  Finishing time: ___________

   Were you able to easily answer this question? Not at all □—□—□—□ Absolutely □—□—□—□ Very clear

4. Must a QuattroFormaggi pizza have only CheeseyTopping toppings(Y/N)? _____

   Start time: ___________  Finishing time: ___________

   Were you able to easily answer this question? Not at all □—□—□—□ Absolutely □—□—□—□ Very clear

5. How many toppings must an InterestingPizza have? ______________________

   Start time: ___________  Finishing time: ___________

   Were you able to easily answer this question? Not at all □—□—□—□ Absolutely □—□—□—□ Very clear

6. List the pairs of atomic sub-concepts of Pizza in which the two concepts are disjoint from each other.
7. What are the toppings that must go on an American pizza?

8. For each of the following roles and attributes, state whether the Pizza concept can participate in it.
   a) hasIngredient: ☐ Yes ☐ No
   b) hasBase: ☐ Yes ☐ No
   c) calories: ☐ Yes ☐ No
   d) depth: ☐ Yes ☐ No

9. How many different kinds of toppings can an American Pizza have at most? Which are they?

10. Can an IceCream have a PizzaTopping as a topping(Y/N)? _____

Start time: ____________ Finishing time: ____________
Was the question clear? Not clear at all ☐☐☐☐☐☐☐☐ Very clear
Were you able to easily answer this question? Not at all ☐☐☐☐☐☐☐☐ Absolutely
Post-test questionnaire

Please answer the following questions at the end of your test.

1. How would you rate the general difficulty of the comprehension tasks of the test?
   Very easy □□□□□ □□□□□ Very difficult

2. How would you rate the difficulty of learning the symbols of the Graphol language?
   Very easy □□□□□ □□□□□ Very difficult

3. How would you rate the general difficulty of using Graphol for reading ontologies?
   Very easy □□□□□ □□□□□ Very difficult

4. (Optional) Aspects of the Graphol language that you particularly like:

   ___________________________________________________________
   ___________________________________________________________
   ___________________________________________________________
   ___________________________________________________________
   ___________________________________________________________
   ___________________________________________________________
   ___________________________________________________________
   ___________________________________________________________

5. (Optional) Aspects of the Graphol language that you would like to see improved/changed:

   ___________________________________________________________
   ___________________________________________________________
   ___________________________________________________________
   ___________________________________________________________
   ___________________________________________________________
   ___________________________________________________________
   ___________________________________________________________
   ___________________________________________________________
Appendix A

Model used for Comprehension Task
# OWLGrEd: a graphical ontology language

## Survey and User Evaluation Study

### Purpose of the Questionnaire

Evaluate the usability of the OWLGrEd language for ontology modeling through a series of comprehension tasks.

### OWLGrEd Cheat Sheet - 1

#### Graphical symbols

The “Concept”, “Role”, “Attribute”, and “Value-domain” labels on the symbols indicate the name of the respective ontology predicate.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
<th>Symbol</th>
<th>Meaning</th>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Concept" /></td>
<td>Concept</td>
<td><img src="image" alt="Disjoint hierarchy connector" /></td>
<td>Disjoint hierarchy connector</td>
<td><img src="image" alt="Restriction edge" /></td>
<td>Restriction edge</td>
</tr>
<tr>
<td><img src="image" alt="Role" /></td>
<td>Role</td>
<td><img src="image" alt="Complete hierarchy connector" /></td>
<td>Complete hierarchy connector</td>
<td><img src="image" alt="Equivalence edge" /></td>
<td>Equivalence edge</td>
</tr>
<tr>
<td><img src="image" alt="Attribute" /></td>
<td>Attribute</td>
<td><img src="image" alt="Inclusion edge" /></td>
<td>Inclusion edge</td>
<td><img src="image" alt="Disjunction edge" /></td>
<td>Disjunction edge</td>
</tr>
<tr>
<td><img src="image" alt="Value domain" /></td>
<td>Value domain</td>
<td><img src="image" alt="Input edge" /></td>
<td>Input edge</td>
<td><img src="image" alt="Complement edge" /></td>
<td>Complement edge</td>
</tr>
<tr>
<td><img src="image" alt="Hierarchy connector" /></td>
<td>Hierarchy connector</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
In the following table you can see some of the more typical examples of conceptual design in OWLGrEd. The “Concept” label on concept nodes in the OWLGrEd column can be replaced with a complex expression label, which indicates that the concept node represents a concept expression such as the union or intersection of concepts. In OWLGrEd, concept inclusion can alternatively be represented through an inclusion label, “<”, followed by the including concept, in the concept node.

<table>
<thead>
<tr>
<th>Relation</th>
<th>Graphol</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISA between concepts</td>
<td><img src="image" alt="Graphol for ISA between concepts" /></td>
</tr>
<tr>
<td>Disjointness between concepts</td>
<td><img src="image" alt="Graphol for Disjointness between concepts" /></td>
</tr>
<tr>
<td>Equivalence between concepts</td>
<td><img src="image" alt="Graphol for Equivalence between concepts" /></td>
</tr>
<tr>
<td>ISA between roles</td>
<td><img src="image" alt="Graphol for ISA between roles" /></td>
</tr>
<tr>
<td>Disjointness between roles</td>
<td><img src="image" alt="Graphol for Disjointness between roles" /></td>
</tr>
<tr>
<td>Equivalence between roles</td>
<td><img src="image" alt="Graphol for Equivalence between roles" /></td>
</tr>
<tr>
<td>ISA between attributes</td>
<td><img src="image" alt="Graphol for ISA between attributes" /></td>
</tr>
<tr>
<td>Disjointness between attributes</td>
<td><img src="image" alt="Graphol for Disjointness between attributes" /></td>
</tr>
<tr>
<td>Equivalence between attributes</td>
<td><img src="image" alt="Graphol for Equivalence between attributes" /></td>
</tr>
<tr>
<td>Typings of the domain and range of a role</td>
<td><img src="image" alt="Graphol for Typings of the domain and range of a role" /></td>
</tr>
<tr>
<td>Typings of the domain and range of an attribute</td>
<td><img src="image" alt="Graphol for Typings of the domain and range of an attribute" /></td>
</tr>
<tr>
<td>Cardinality of roles/attributes</td>
<td><img src="image" alt="Graphol for Cardinality of roles/attributes" /></td>
</tr>
<tr>
<td>Restriction on roles/attributes</td>
<td><img src="image" alt="Graphol for Restriction on roles/attributes" /></td>
</tr>
<tr>
<td>Functionality assertions</td>
<td><img src="image" alt="Graphol for Functionality assertions" /></td>
</tr>
</tbody>
</table>

Restriction-type = some/only/exactly x/max x/min x
Comprehension Task

You have been provided a printout of a simple OWLGrEd model which depicts an extract of the popular Pizza ontology (also see Appendix A). The model essentially represents the fact that a pizza is composed of a base and one or more toppings, and shows how different toppings are matched with different kinds of pizza.

Please answer the following questions.

1. List the typings of the domain and range of the following roles.
   a) hasIngredient Domain: __________________ Range: __________________
   b) hasTopping Domain: __________________ Range: __________________
   c) hasBase Domain: __________________ Range: __________________

   Start time: ____________    Finishing time: ____________

   Was the question clear? Not clear at all □—□—□—□—□ Very clear
   Were you able to easily answer this question? Not at all □—□—□—□—□ Absolutely

2. Must a Napoletana Pizza have a SpicyTopping(Y/N)? _____

   Start time: ____________    Finishing time: ____________

   Was the question clear? Not clear at all □—□—□—□—□ Very clear
   Were you able to easily answer this question? Not at all □—□—□—□—□ Absolutely

3. Specify whether the following roles and attributes are functional, inverse functional, or both.
   a) hasIngredient: □ Functional □ Inverse functional
   b) hasBase: □ Functional □ Inverse functional
   c) calories: □ Functional □ Inverse functional
   d) depth: □ Functional □ Inverse functional

   Start time: ____________    Finishing time: ____________

   Was the question clear? Not clear at all □—□—□—□—□ Very clear
   Were you able to easily answer this question? Not at all □—□—□—□—□ Absolutely

4. Must a QuattroFormaggi pizza have only CheeseyTopping toppings(Y/N)? _____

   Start time: ____________    Finishing time: ____________

   Was the question clear? Not clear at all □—□—□—□—□ Very clear
   Were you able to easily answer this question? Not at all □—□—□—□—□ Absolutely

5. How many toppings must an InterestingPizza have? ________________________

   Start time: ____________    Finishing time: ____________

   Was the question clear? Not clear at all □—□—□—□—□ Very clear
   Were you able to easily answer this question? Not at all □—□—□—□—□ Absolutely

6. List the pairs of direct sub-concepts of Pizza in which the two concepts are disjoint from each other.

   ________________________

   Start time: ____________    Finishing time: ____________

   Was the question clear? Not clear at all □—□—□—□—□ Very clear
   Were you able to easily answer this question? Not at all □—□—□—□—□ Absolutely
7. What are the toppings that must go on an American pizza?

Start time: ____________  Finishing time: ____________
Was the question clear? Not clear at all ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ Very clear
Were you able to easily answer this question? Not at all ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ Absolutely

8. For each of the following roles and attributes, state whether the Pizza concept can participate in it.
   a) hasIngredient: ☐ Yes ☐ No
   b) hasBase: ☐ Yes ☐ No
   c) calories: ☐ Yes ☐ No
   d) depth: ☐ Yes ☐ No

Start time: ____________  Finishing time: ____________
Was the question clear? Not clear at all ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ Very clear
Were you able to easily answer this question? Not at all ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ Absolutely

9. How many different kinds of toppings can an American Pizza have at most? Which are they?

Start time: ____________  Finishing time: ____________
Was the question clear? Not clear at all ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ Very clear
Were you able to easily answer this question? Not at all ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ Absolutely

10. Can an IceCream have a PizzaTopping as a topping(Y/N)? _____
    Start time: ____________  Finishing time: ____________
    Was the question clear? Not clear at all ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ Very clear
    Were you able to easily answer this question? Not at all ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ — ☐ Absolutely
Post-test questionnaire

Please answer the following questions at the end of your test.

1. **How would you rate the general difficulty of the comprehension tasks of the test?**
   Very easy [ ]—[ ]—[ ]—[ ]—[ ]— Very difficult

2. **How would you rate the difficulty of learning the symbols of the OWLGrEd language?**
   Very easy [ ]—[ ]—[ ]—[ ]—[ ]— Very difficult

3. **How would you rate the general difficulty of using OWLGrEd for reading ontologies?**
   Very easy [ ]—[ ]—[ ]—[ ]—[ ]— Very difficult

4. (Optional) Aspects of the OWLGrEd language that you particularly like:
   ____________________________________________________________
   ____________________________________________________________
   ____________________________________________________________
   ____________________________________________________________
   ____________________________________________________________
   ____________________________________________________________
   ____________________________________________________________
   ____________________________________________________________
   ____________________________________________________________
   ____________________________________________________________

5. (Optional) Aspects of the OWLGrEd language that you would like to see improved/changed:
   ____________________________________________________________
   ____________________________________________________________
   ____________________________________________________________
   ____________________________________________________________
   ____________________________________________________________
   ____________________________________________________________
   ____________________________________________________________
   ____________________________________________________________
   ____________________________________________________________
Appendix A

Model used for Comprehension Task
Graphol: a graphical ontology language
Survey and User Evaluation Study

Purpose of the Questionnaire
Evaluate the usability of the Graphol language for both comprehension and editing.

Pre-questionnaire
We would like for you to provide some personal and professional background information, regarding experience with ontology modelling, knowledge representation, and ontology languages (OWL2, Description Logics). Please fill in the following short survey.

1. Your name: ____________________________

2. How old are you? _____

3. Please describe your occupation/profession and role?

4. What is type and field of your highest completed education degree?

5. Roughly how many years of experience with ontologies do you have (0 if none)? ______

6. On a scale from 1 to 5, how would you rate your knowledge of conceptual modeling?
   None — — — — Very high

7. On a scale from 1 to 5, how would you rate your knowledge of ontologies?
   None — — — — Very high

8. Which ontology editors are you familiar with?
   □ Protégé □ TopBraid Composer □ NeOn Toolkit □ OWLGrEd
   □ OntoStudio □ OntoUML □ Other:

9. Which knowledge representation and conceptual modeling formalisms are you familiar with?
   □ OWL2 □ E-R □ Description Logics □ UML Class Diagrams
   □ ORM □ Other:

10. Have you ever had experience in manually creating or editing an ontology (Y/N)? ______

11. Have you ever had experience in working with medium- or large-scale ontologies in real-life projects (Y/N)? ______

12. Which ontology visualization tools are you familiar with?
   □ OntoGraf □ OWLViz □ GROWL □ OWLGrEd □ Other:
## Graphol Cheat Sheet - 1

### Graphical symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
<th>Symbol</th>
<th>Meaning</th>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Concept" /></td>
<td>Concept</td>
<td><img src="image" alt="Intersection" /></td>
<td>Intersection</td>
<td><img src="image" alt="Domain restriction" /></td>
<td>Domain restriction</td>
</tr>
<tr>
<td><img src="image" alt="Role" /></td>
<td>Role</td>
<td><img src="image" alt="Union" /></td>
<td>Union</td>
<td><img src="image" alt="Restriction" /></td>
<td>Restriction</td>
</tr>
<tr>
<td><img src="image" alt="Attribute" /></td>
<td>Attribute</td>
<td><img src="image" alt="Complement" /></td>
<td>Complement</td>
<td><img src="image" alt="Inclusion edge" /></td>
<td>Inclusion edge</td>
</tr>
<tr>
<td><img src="image" alt="Value domain" /></td>
<td>Value domain</td>
<td><img src="image" alt="Inverse" /></td>
<td>Inverse</td>
<td><img src="image" alt="Parameter input edge" /></td>
<td>Parameter input edge</td>
</tr>
<tr>
<td><img src="image" alt="Disjoint Union" /></td>
<td>Disjoint Union</td>
<td><img src="image" alt="Role chain" /></td>
<td>Role chain</td>
<td><img src="image" alt="Global functionality edge" /></td>
<td>Global functionality edge</td>
</tr>
<tr>
<td><img src="image" alt="Enumeration" /></td>
<td>Enumeration</td>
<td><img src="image" alt="Restriction type" /></td>
<td>Restriction type</td>
<td><img src="image" alt="Restriction type" /></td>
<td>Restriction type</td>
</tr>
</tbody>
</table>

### Restriction types:

- **existential**: “exists”
- **universal**: “forall”
- **cardinality**: “x,y”, with $y \geq x$, and $x = \text{"\text{-}"}$ if no minimum cardinality is specified, and $y = \text{"\text{-}"}$ if no maximum cardinality is specified

If the restriction type label on a white or black square is missing, then the existential restriction is implied.
Graphol Cheat Sheet - 2

Graphol intentional assertions

In the following table, the “Concept” and “Role” nodes in the Graphol column can be replaced with complex concepts and roles.

<table>
<thead>
<tr>
<th>Graphol Expression</th>
<th>Meaning</th>
<th>Graphol Expression</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Concept inclusion</td>
<td></td>
<td>Role inclusion</td>
</tr>
<tr>
<td></td>
<td>Attribute inclusion</td>
<td></td>
<td>Concept equivalence</td>
</tr>
<tr>
<td></td>
<td>Role equivalence</td>
<td></td>
<td>Attribute equivalence</td>
</tr>
<tr>
<td></td>
<td>Globally functional role</td>
<td></td>
<td>Globally inverse functional role</td>
</tr>
<tr>
<td></td>
<td>Globally functional and inverse</td>
<td></td>
<td>Globally functional attribute</td>
</tr>
</tbody>
</table>

Each concept or role expression is a tree whose edges are exclusively parameter input edges or global functionality edges. The root of the tree is the “most external” operator. The inclusion arrow connects two such roots.

Graphol expressions

In the following tables we provide the most common expressions. N-ary operators with N greater than 2 are represented as having three input parameters. Restrictions for role domains exist obviously also for role ranges.

<table>
<thead>
<tr>
<th>Graphol Expression</th>
<th>Meaning</th>
<th>Graphol Expression</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Concept</td>
<td></td>
<td>Concept intersection</td>
</tr>
<tr>
<td></td>
<td>Concept union</td>
<td></td>
<td>Concept union</td>
</tr>
<tr>
<td></td>
<td>Universal role domain</td>
<td></td>
<td>Universal attribute domain</td>
</tr>
<tr>
<td></td>
<td>Existential role domain</td>
<td></td>
<td>Existential attribute domain</td>
</tr>
<tr>
<td></td>
<td>Max. and min. cardinality role</td>
<td></td>
<td>Max. and min. cardinality</td>
</tr>
<tr>
<td></td>
<td>Value enumeration</td>
<td></td>
<td>Role</td>
</tr>
<tr>
<td></td>
<td>Inverse role</td>
<td></td>
<td>Role chain</td>
</tr>
<tr>
<td></td>
<td>Attribute</td>
<td></td>
<td>Attribute</td>
</tr>
</tbody>
</table>
Graphol Cheat Sheet - 3

Axiom examples

In the following table we show some common logical axiom examples and their equivalent Graphol expression.

<table>
<thead>
<tr>
<th>Graphol Expression</th>
<th>Logical axiom</th>
<th>Graphol Expression</th>
<th>Logical axiom</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Disjoint concepts</td>
<td></td>
<td>Inverse roles</td>
</tr>
<tr>
<td></td>
<td>Role domain typing</td>
<td></td>
<td>Role range typing</td>
</tr>
<tr>
<td></td>
<td>Attribute domain typing</td>
<td></td>
<td>Attribute range typing</td>
</tr>
<tr>
<td></td>
<td>Global role functionality</td>
<td></td>
<td>Global inverse role functionality</td>
</tr>
<tr>
<td></td>
<td>Global attribute functionality</td>
<td></td>
<td>Role reflexivity</td>
</tr>
<tr>
<td></td>
<td>Role symmetry</td>
<td></td>
<td>Role transitivity</td>
</tr>
</tbody>
</table>
Comprehension Task

You have been provided a printout of a simple Graphol model which depicts an extract of the popular Pizza ontology (also see Appendix A). The model essentially represents the fact that a pizza is composed of a base and one or more toppings, and shows how different toppings are matched with different kinds of pizza.

Please answer the following questions.

1. List the typings of the domain and range of the following roles.

   a) hasIngredient  Domain: ___________________  Range: ___________________
   b) hasTopping     Domain: ___________________  Range: ___________________
   c) hasBase        Domain: ___________________  Range: ___________________

   Start time: ___________  Finishing time: ___________

   Was the question clear? Not clear at all □—□—□—□—□ Very clear
   Were you able to easily answer this question? Not at all □—□—□—□—□ Absolutely

2. Specify whether the following roles and attributes are globally functional, globally inverse functional, or both.

   a) hasIngredient: □ Functional  □ Inverse functional
   b) hasTopping:     □ Functional  □ Inverse functional
   c) hasBase:        □ Functional  □ Inverse functional
   d) calories:       □ Functional  □ Inverse functional
   e) depth:          □ Functional  □ Inverse functional

   Start time: ___________  Finishing time: ___________

   Was the question clear? Not clear at all □—□—□—□—□ Very clear
   Were you able to easily answer this question? Not at all □—□—□—□—□ Absolutely

3. How many toppings must an InterestingPizza have? _________

   Start time: ___________  Finishing time: ___________

   Was the question clear? Not clear at all □—□—□—□—□ Very clear
   Were you able to easily answer this question? Not at all □—□—□—□—□ Absolutely

4. Two sub-concepts of Pizza are equivalent to complex concepts. List which sub-concepts they are and describe in natural language or in any OWL2 syntax what they are equivalent to.

   a) ________________________________
      Equivalent to: ________________________________

   b) ________________________________
      Equivalent to: ________________________________

   Start time: ___________  Finishing time: ___________

   Was the question clear? Not clear at all □—□—□—□—□ Very clear
   Were you able to easily answer this question? Not at all □—□—□—□—□ Absolutely
5. List the pairs of atomic sub-concepts of Pizza in which the two concepts are disjoint from each other.

6. What are the toppings that must go on a Diavola pizza?

7. What are the toppings that can go on a Diavola pizza?

8. For each of the following roles and attributes, state whether the Pizza concept can participate in it, and, if so, specify if the participation is mandatory.
   a) hasIngredient: □ Yes □ No □ Mandatory
   b) hasBase: □ Yes □ No □ Mandatory
   c) calories: □ Yes □ No □ Mandatory
   d) depth: □ Yes □ No □ Mandatory

9. What role is transitive in the model? ________________________

10. The model defines which Country is the country of origin for two kinds of Pizza. Which are they and what is their Country of origin?
   a) ________________________ Country of origin: ________________________
   b) ________________________ Country of origin: ________________________

Was the question clear? Not clear at all □—□—□—□—□ Very clear
Were you able to easily answer this question? Not at all □—□—□—□—□ Absolutely
Editing Task

On your screen you see a simple Graphol model which depicts some of the relations that can occur between persons in a family (also see Appendix B). Please use the provided Graphol palette for the yEd editor to perform the tasks described below. Feel free to repeat, if necessary, the concept, role, attribute or operator symbols in the ontology, and do not concern yourself with layout issues in the model, just worry about the syntactic and semantic correctness of the changes or additions to the model.

1. Model the fact that the domain of the hasChild role is either a Father or a Mother.

Start time: ____________  Finishing time: ____________

Was it clear to you how to perform this task? Not clear at all □—□—□—□—□ Very clear
Were you able to quickly and easily perform this task? Not at all □—□—□—□—□ Absolutely

2. Add the following attributes, with the listed properties, to the Person concept.
   a) name
      • Domain: Person
      • Range: xsd:string
      • Person has mandatory participation
      • Globally functional
   b) telephoneNumber
      • Domain: Person
      • Range : xsd:string

Start time: ____________  Finishing time: ____________

Was it clear to you how to perform this task? Not clear at all □—□—□—□—□ Very clear
Were you able to quickly and easily perform this task? Not at all □—□—□—□—□ Absolutely

3. Model the fact that the hasChild role is inverse of the hasParent role.

Start time: ____________  Finishing time: ____________

Was it clear to you how to perform this task? Not clear at all □—□—□—□—□ Very clear
Were you able to quickly and easily perform this task? Not at all □—□—□—□—□ Absolutely

4. The hasParent role is used to link a Person with another Person which is his parent. Model the fact that a Person has exactly two parents.

Start time: ____________  Finishing time: ____________

Was it clear to you how to perform this task? Not clear at all □—□—□—□—□ Very clear
Were you able to quickly and easily perform this task? Not at all □—□—□—□—□ Absolutely

5. Model the fact that the range of the hasSister role is not a Male.

Start time: ____________  Finishing time: ____________

Was it clear to you how to perform this task? Not clear at all □—□—□—□—□ Very clear
Were you able to quickly and easily perform this task? Not at all □—□—□—□—□ Absolutely
6. Model the fact that each Father must have a child and that each Mother must have a child.

Start time: ____________  Finishing time: ____________
Was it clear to you how to perform this task? Not clear at all □□□□□ Very clear
Were you able to quickly and easily perform this task? Not at all □□□□□ Absolutely

7. Model the fact that both hasFather and hasMother are functional roles.

Start time: ____________  Finishing time: ____________
Was it clear to you how to perform this task? Not clear at all □□□□□ Very clear
Were you able to quickly and easily perform this task? Not at all □□□□□ Absolutely

8. Model the fact that a Non-european Person cannot have citizenship in a European Nation.

Start time: ____________  Finishing time: ____________
Was it clear to you how to perform this task? Not clear at all □□□□□ Very clear
Were you able to quickly and easily perform this task? Not at all □□□□□ Absolutely

9. Model the fact that hasAncestor is a transitive role.

Start time: ____________  Finishing time: ____________
Was it clear to you how to perform this task? Not clear at all □□□□□ Very clear
Were you able to quickly and easily perform this task? Not at all □□□□□ Absolutely

10. Model the fact that hasSibling is a symmetric role.

Start time: ____________  Finishing time: ____________
Was it clear to you how to perform this task? Not clear at all □□□□□ Very clear
Were you able to quickly and easily perform this task? Not at all □□□□□ Absolutely
Post-test questionnaire

Please answer the following questions at the end of your test.

1. How would you rate the general difficulty of the comprehension tasks of the test?
   Very easy □□□□□ Very difficult

2. How would you rate the general difficulty of the editing tasks of the test?
   Very easy □□□□□ Very difficult

3. How would you rate the difficulty of learning the symbols of the Graphol language?
   Very easy □□□□□ Very difficult

4. How would you rate the general difficulty of using Graphol for reading or editing ontologies?
   Very easy □□□□□ Very difficult

5. Would you use Graphol for editing ontologies?
   Absolutely □□□□□ Not at all

6. (Optional) Aspects of the Graphol language that you particularly like:

   __________________________________________
   __________________________________________
   __________________________________________
   __________________________________________
   __________________________________________
   __________________________________________
   __________________________________________

7. (Optional) Aspects of the Graphol language that you would like to see improved/changed:

   __________________________________________
   __________________________________________
   __________________________________________
   __________________________________________
   __________________________________________
   __________________________________________
Appendix A

Model used for Comprehension Task
Appendix B

Model used for Editing Task