Explanatory notes on the OpenGALEN OWL Translations

Table of Contents

License

This document, and the OWL translations it accompanies, are provided under the GALEN Open Source License (GOSL) version 1.0, a copy of which is presented below:

The GALEN Open Source License (GOSL)

Version 1.0 11th October 1999

The OpenGALEN Clinical Terminology is Copyright © the University of Manchester, UK, the University of Nijmegen, NL, and their contributors.

'Source Material' refers to:

- text files containing Grail source code, GALEN Intermediate Representation dissections, GALEN Intermediate Representation configuration files, or linguistic or other mappings relating to the OpenGALEN Clinical Terminology, AND which contain the following (or substantially similar) notice:

'The contents of this file are Copyright © the Universities of Manchester and Nijmegen and are covered by the GALEN Open Source License (GOSL), a copy of which should be found as part of the package within which this file was found, or which can be obtained from www.opengalen.org. You are free to use, modify and re-distribute this material, as long as this notice remains intact, and as long as any modifications you make are clearly indicated.'

OR

- binary data-sets containing material as described above (which could be represented in textual form), but which are readable by commonly available database or spreadsheet applications, and which have as part of their description or comment the same notice as described above

'Compiled form' refers to:

- Binary representations which result from the application of the functionality of a GALEN Terminology Server on Source Material. For a description of the GALEN Terminology Server, see www.opengalen.org.

'OpenGALEN Clinical Terminology' refers to:

- Source Material and/or Compiled form (or substantive derivatives) licensed through OpenGALEN (www.opengalen.org). The arbiter of substantive derivative shall be OpenGALEN.

Redistribution of Source Material, and its use in Source Material and Compiled forms, with or without modification, are permitted provided that the following conditions are met:

1. Re-distributions of Source Material must retain the whole text of this GOSL (or, at your choice, any minor-version updates of this GOSL numbered 1.x published at www.opengalen.org) verbatim (including Disclaimer) in a prominently visible file accompanying such distribution, and each individual file must retain its copyright notice as described in the definition of 'Source Material' above.

2. Modifications of Source Material must retain the original copyright notice in each file, together with a description of what modifications were made, when, and by whom.

3. Re-distributions in Compiled form must reproduce the whole text of this GOSL (or, at your choice, any minor-version updates of this GOSL numbered 1.x published at www.opengalen.org) verbatim (including Disclaimer) in the documentation and/or other materials provided with the distribution. Re-distribution in Compiled form must be accompanied by the Source Material from which the Compiled form came about.

4. All advertising materials of any product mentioning features or use of the OpenGALEN Clinical Terminology must display the following acknowledgement, which must also be prominently visible within, or as part of the start-up of, any application that uses the OpenGALEN Clinical Terminology at run-time, or which has benefited from it use during build/development-time:

This product uses the OpenGALEN Clinical Terminology, Copyright the Universities of Manchester and Nijmegen, licensed through OpenGALEN (www.opengalen.org).

5. Neither name of the University of Manchester or Nijmegen, nor the names of its contributors, nor that of OpenGALEN may be used to endorse or promote products without specific prior written permission.

Disclaimer:

THIS MATERIAL IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS ``AS IS'' AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS MATERIAL, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Introduction

This document accompanies the OWL translations of OpenGALEN ontology components.

The OpenGALEN Common Reference Model (CRM) is a rich compositional ontology of the medical domain, covering anatomy, function, pathology, diseases, symptoms, drugs and procedures. Original sources are expressed in the GRAIL description logic language and these, together with associated documentation, selected academic papers and other references are available from www.opengalen.org.

Translation to OWL 2.0 of the GRAIL sources was achieved by adapting a native GRAIL compiler (in the OpenKnoME tool) so that, during its incremental compilation of the whole CRM content, individual GRAIL statements were also serially copied out to file as OWL RDF expressions.

Copies of OpenKnoME 5.4d (available fromwww.opengalen.org/sources/software.html) can be adapted in the same way by applying the GROWL patch[1], also available from the same site.

The translation is necessarily approximate; certain GRAIL language constructs and GRAIL source meta-constructs have no exact OWL 2.0 equivalent:

GRAIL Cardinality and Functional Roles
GRAIL Sanctioning
GRAIL Source Model Hierarchy

Details of how these constructs have been handled in the translation are provided in this document. Particular attention should be paid to the translation of GRAIL Cardinality.

Translation of selected GRAIL constructs

As a partial guide, Appendix A to this document shows a small GALEN ontology together with the OWL RDF translation of the same ontology; the following sections provide further detail concerning specific aspects of the translation approach.

SpecialisedBy

All GRAIL 'specialisedBy' statements used to construct the OpenGALEN Common Reference Model model (CRM) are translated into OWL propertyChain axioms. This is believed to be an exact translation. However, the set of propertyChain axioms used in the CRM is known to include cycles such that the ontology as a whole can not be classified by current DL classifiers (October 2009). The native GALEN classifier uses a structural algorithm which is correct but incomplete for ontologies that include property chains (but both correct *and* complete if no property chains exist in the ontology).

In the original GRAIL sources - and therefore also in the original RDF serialized translation - all the relevant property chain statements are collected in a block toward the start of the ontology. In the OWL-RDF translation, therefore, they can easily be commented out if necessary. However this very significantly diminishes both the computational challenge and the usefulness of the ontology.

[1] From the KnomePro console, select menu option 'Special…Apply patch'. Navigate to the directory containing the GROWL patch and click 'apply patch'. Emptying the change set after patch application is optional.

Hierarchy Dump Files

The difference in the classification brought about by the inclusion of the full specialisedBy/property chain set may be determined by examination of the 'Hierarchy Dump Files' included with some OWL translations. These files list all the direct parent-child subsumption relationships that hold between all classes in the final 'inferred view' as derived by GALEN's incremental classifier.

Each full OpenGALEN translation release includes one hierarchy dump for the CRM when compiled with all property chains included and another hierarchy dump for the same model but with the property chains omitted. The Dublin Core <description> element for the OWL RDF translation will also record how long each of these two classification processes took on a reference machine both with, and without, property chains.

Sanctioning

GRAIL sanctioning statements ('grammatical' and 'sensible') are represented in this translation as annotations since, although they comprise a significant part of the knowledge content of the CRM proper, they are closed world statements without implication and thus are not equivalent to OWL domain and range restrictions.

Briefly, if there exists sanctions:

DomainClassG grammatically Role4 RangeClassG.
DomainClassS sensibly Role4 RangeClassS.

..and we then consider the 'Role3' arc within the expression:

```
ClassA which Role1 (ClassB which <
                        Role2 ClassC
                        Role3 (ClassD which < Role4 ClassE Role5 ClassF>))
```

..then the expression as a whole is deemed invalid unless it can be demonstrated that DomainClassG and DomainClassS both subsume or are equal to:

```
ClassB which <
    Role2 ClassC
    Role3 (ClassD which < Role4 ClassE Role5 ClassF>)
```

..and simultaneously that RangeClassG and RangeClassS both subsume or are equal to

```
(ClassD which < Role4 ClassE Role5 ClassF>)
```

Note, however, that to reduce the size of the OWL-RDF file, only the left hand class X in any original GRAIL statement of the form (X grammaticallyOrSensibly role Y) carries the annotation. GRAIL sanctioning semantics are symmetric: (X grammaticallyOrSsensibly att Y) implies (Y grammaticallyOrSensibly invAtt X).

Thus, the GRAIL statement:

ClassA grammatically SomeRole ClassB

becomes, in OWL RDF:

```
<owl:AnnotationProperty rdf:about="#G-SANCTION-SomeRoleAP"/>
<owl:Class rdf:about="#ClassA">
        <G-SANCTION-SomeRoleAP>ClassB</G-SANCTION-SomeRoleAP>
</owl:Class>
```

and

ClassA sensibly SomeRole ClassB

becomes:

```
<owl:AnnotationProperty rdf:about="#S-SANCTION-SomeRoleAP"/>
<owl:Class rdf:about="#ClassA">
        <S-SANCTION-SomeRoleAP>ClassB</S-SANCTION-SomeRoleAP>
</owl:Class>
```

Note that an artifact of this translation is numerous re-declarations of the relevant flavours of annotation property: although it would have been possible to do so, the GRAIL to OWL translation implementation does not build a cumulative dictionary of all annotation properties created so far during serial compilation. Therefore, it does not know when a required annotation property has already been declared and thus is required to (re-)declare them each time they are needed. This approach was chosen because OWL implementations are tolerant of this redundancy, and resource limitations did not allow for a neater implementation.

In theory, GRAIL grammatical sanctions might be represented as universal (only) restrictions plus disjointness axioms between all classes in the OpenGALEN Top Ontology, but such disjointness axioms are not an explicit part of the GRAIL source and would therefore need to be computed. For this reason this translation approach was not explored.

Extrinsically

In addition to GRAIL sanctions (outlined above) and necessary statements (implemented as existential restrictions), the GRAIL language also supports a form of annotation property in which any pairwise combination of classes, strings or integer value types may be associated via *any* flavour of semantic link. Such links are always bidirectional.

GRAIL does not differentiate between ontology roles and extrinsic roles, so in theory it allows the concurrent use of a single ‘part_of’ role both within restrictions modeling true partonomy (with implication and inheritance) and, simultaneously, in extrinsic statements on classes in the same ontology in order to attach e.g. the URLs to a library of relevant anatomical images, and thus without implication or inheritance.

In the translation (see also Appendix A), extrinsics statements are mapped to OWL annotation properties, involving the declaration of a new annotation property whose name is postfixed with ‘AP’:

```
<!-- ClassA extrinsically RoleA ClassB. -->
<owl:AnnotationProperty rdf:about="#RoleAAP"/>
<owl:Class rdf:about="#ClassA">
        <RoleAAP>ClassB</RoleAAP>
</owl:Class>
```

which / whichG

GRAIL has two levels of sanctioning (grammatical and sensible), and thus also two different constructors (which and whichG) to indicate which level must be satisfied by each individual arc within every candidate expression.

Thus, if:

A newSub [B C D E].
A grammatically R B.
B grammatically R C.
A grammaticallyAndSensibly R D.

then the candidate expression:

(A whichG R E)

… is rejected as ill-formed; the sanctioning does NOT allow inference that E is a subclass of (B or D). Similarly, both candidate expressions:

(A which R B)
(A whichG R (B which R C)

…are also rejected, since there is no sensible level of sanctioning in play for all arcs. But all of:

(A whichG R B)
(A whichG R (B whichG R C)
(A whichG R D)
(A which R D)

..will pass the sanctioning check and so will be reified *permanently* as a whole within the incrementally classified inferred view.

Note, however, that executing sanctioning checks on *each* arc of any candidate expression requires that many subgraphs of the expression must first be classified and reified so that all inheritable sanctions can be found that should apply to the domain and range of the arc in question. This process largely explains why the inferred view hierarchy dumps contain many anonymous classes not found in the original GRAIL sources.

Within the OWL translation, which level of sanctioning applies to a given arc is represented by means of explicitly encoding only the whichG constructors, leaving the which constructors implicit: where an arc within a candidate expression graph is required to satisfy only grammatical sanctioning (ie where the whichG operator was used in the original GRAIL source) then the relevant subgraph of the whole candidate is exported as a separate class and with a machine constructed name, and an OWL annotation is placed on that class indicating that all restriction arcs radiating directly out from the base class are required to satisfy only the grammatical level of sanctioning. The full original candidate expression is then exported later with the relevant subgraph(s) incorporated by reference to the artificially created class.:

```
<!-- (ClassA which RoleA (ClassB whichG RoleA ClassC) )name ClassD. -->
<owl:Class rdf:about="#ClassBwhichRoleAClassC">
    <owl:equivalentClass>
        <owl:Class>
            <owl:intersectionOf rdf:parseType="Collection">
                <rdf:Description rdf:about="#ClassB"/>
                <owl:Restriction>
                    <owl:onProperty rdf:resource="#RoleA"/>
                    <owl:someValuesFrom rdf:resource="#ClassC"/>
                </owl:Restriction>
            </owl:intersectionOf>
        </owl:Class>
    </owl:equivalentClass>
</owl:Class>

<owl:AnnotationProperty rdf:about="#SANCTION-LEVELAP"/><owl:Class
rdf:about="#ClassBwhichRoleAClassC">
    <SANCTION-LEVELAP rdf:datatype="&xsd;string">GRAMMATICAL</SANCTION-LEVELAP>
</owl:Class>

<owl:Class rdf:about="#ClassD">
    <owl:equivalentClass>
        <owl:Class>
            <owl:intersectionOf rdf:parseType="Collection">
                <rdf:Description rdf:about="#ClassA"/>
                <owl:Restriction>
                    <owl:onProperty rdf:resource="#RoleA"/>
                    <owl:someValuesFrom>
                        <owl:Class rdf:about="#ClassBwhichRoleAClassC"/>
                    </owl:someValuesFrom>
                </owl:Restriction>
            </owl:intersectionOf>
        </owl:Class>
    </owl:equivalentClass>
</owl:Class>
```

This somewhat involved construct is required because OWL2 does not provide a means to directly annotate individual restrictions within a candidate expression graph. However, this translation approach has a side effect of (partially) forcing a simulation within an OWL classification environment of the additional classification effort required to implement GRAIL sanctioning for real, because it forces the reification of many of the subgraph classes involved.

Name

The OWL translation includes large numbers of class equivalence statements. Many (usually recognisably machine generated) are necessary artefacts of the translation (e.g. consequences of the whichG translation, above), whilst others reflect the fact that the GRAIL name operator legitimately allows individual classes to have any number of different accessor strings/knowledge names, as long as each accessor string is unique for one concept.

Note also that the raw OWL-RDF translation files include many redundant duplicate class and object property declaration statements. These are all necessary artefacts of the translation to OWL; they do not reflect similar levels of redundancy in the original GRAIL source.

Cardinality

The translation of GRAIL single valued role cardinality to OWL functional attributes is only a partial translation and, as it stands, is more incorrect than correct; GRAIL cardinality statements are a test rather than an implication (similar to GRAIL sanctioning). The closest complete translation to OWL would require additional explicit statements that all primitive classes are mutually disjoint, but this is not represented in this translation.

The decision to represent GRAIL cardinality as functional roles was taken predominantly because this appears to be the simplest way to represent that information at all. For most classification purposes it would however almost certainly be more appropriate to regard **all** attributes as non-functional even where the translation says they *are* functional. This would also remove the problem that most DL classifiers refuse to reason over ontologies including roles that are both functional and involved in a propertyChain. Future researchers into the value of cardinality as check rather than as implication can extract the cardinality information from the translation and devise more appropriate places to represent and reason over it, presumably outside the DL fragment.

Functional Transitive Roles

The OpenGALEN ontology includes a number of roles that are simultaneously transitive and functional (or inverse functional). This combination of properties within is not allowed in OWL/OWL2 and is normally assumed to indicate a modelling error. Such roles appear in the OpenGALENontology because the GRAIL logic interprets the construct somewhat differently by comparison with mainstream DLs:

IF

p is a transitive role,

and

A p some B, B p some C

and

to say that p manyOne

THEN this implied that

(p Some C) subsumed (p some B)

and as a consequence,

IF A p some B AND A p some D)

THEN

EITHER B p some D

OR EXISTS C . B p some C AND C p some D.

(The original definitions were more verbose concerning the recursion)

The point is that the combination of 'manyOne' and 'transitive' applied to a property indicates that it is constrained to form an (inverse) tree rather than an acyclic directed graph, that is that for every Entity, there can be only one direct p successor, i.e. each entity can be part of only one branch of the tree defined by the transitive relation "p" - call it the "p-tree".

The direction, "looking up" the "p-tree" may seem backwards, but is usually used for things such as "is-part-of", where the inverse "has-part" formed a strict tree.

(Obviously this could equally well have been written hasPart isPartOf oneMany. hasPart Transitive. or isPartOf hasPart manyOne. isPartOf Transitive. However, OWL cannot express this constraint. The closest one can come is isPartOf transitive isPartOfDirectly subPropertyOf isPartOf. isPartOfDirectly functional.)

In practice this OWL approximation was what was actually often done in GRAILmodelling, but the language supported the alternative construct and there are therefore some properties that are **both** transitive **and** given a cardinality of manyOne or oneMany, which would otherwise be translated to OWL as "functional" or "inverseFunctional". The implication is that the translation of GRAIL many:one and one:many cardinality to the OWL functional property is not correct for GRAIL attributes that are also transitive, although this is what is in fact stated in the current translation.

The simplest fix would be to post-process the currently published OWL translations so as to re-translate each role r found to be both transitivand functional role into a pair of roles:

rTransitive Transitive.
rFunctional subPropertyOf rTransitive.
rFunctional Functional.

Then, refactor all of the current translations of the following GRAIL patterns involving role r as shown

ORIGINAL GRAIL	CORRECTED OWL
C topicNecessarily r D	C --> rFunctional some D
(C which r D) name CrD	CrD <--> C that rTransitive some D
(C whichG r D) name CrD	CrD <--> C that rTransitive some D
X topicNecessarily (Q which r D)	X --> Q that rTransitive some D.

(see Appendix A for how these statements are currently translated to OWL)

…with corresponding fixes of any statements involving inverse-r.

There might be a few special cases missed, but those really should have had rDirectly in the GRAIL.The re-write captures the intent at the cost of some meta reasoning about the usage context but would avoid any actual violations of OWL rules.

Jan 2010: Yevgeny Kazakov has raised concerns about the intended GRAIL semantics. The logic may be non-monotonic, in that subsumptions can disappear after the addition of further axioms

For example, if we have just axioms:

hasPart isPartOf oneMany
A hasPart B
B isPartOf C

then this implies that

C subsumes A

On the other hand, as soon as we add the transitivity axiom:

hasPart Transitive

Then this subsumption is lost since C can now be "in between" A and B.

The logic is also likely to be non-deterministic, namely if we have many statements like

A isPartOf B1
A isPartOf B2
...
A isPartOf Bn

then there are exponentially many configurations possible of how B1,...,Bn are included into each other.

Source file hierarchy

GALEN-native classifiers, although incremental, require declaration of new classes before they can be used in restrictions. Thus, successful compilation of the original GRAIL sources is statement-order dependent: a single continuous stream of statements is incrementally compiled statement by statement but the elements of that stream can not be compiled in random order.

Partly because of this issue and partly because of the sheer size of the GRAIL source code for the CRM, this statement stream is in practice fragmented across more than a thousand individually versioned source units of varying size, with the whole being organised into a monohierarchy to aid navigation around the sources. Reconstruction of the single compile stream is then achieved by a recursive top down tree walk of the same source unit hierarchy. This componentization of the ontology also provides support for collaborative authoring, where branches of the ontology can be checked out to individual authors. Figure 1 shows (at left) the model source unit hierarchy partly expanded and (at bottom right) part of the GRAIL statement content in the highlighted model unit:

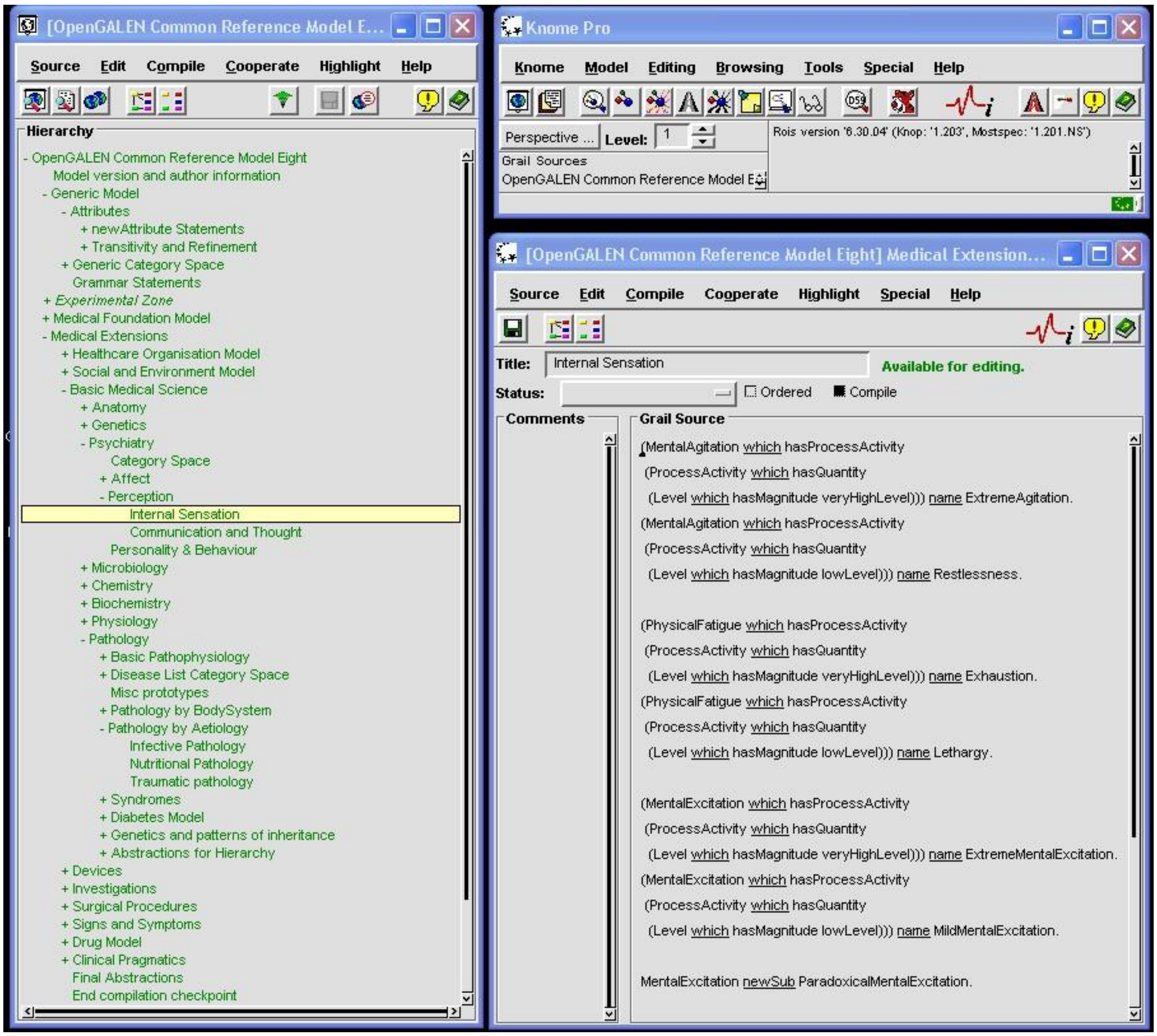

Figure 1: GRAIL Source Unit Hierarchy

This original model source unit componentization is represented in the original OWL RDF export (but may be lost from subsequent transformations) as a series of embedded XML comments indicating the start of each successively compiled source unit. A single XML comment at the end of the RDF export represents the relative monohierarchical arrangement of all source units so compiled.

SELECTED REFERENCES:

A comprehensive list of references can be obtained from http://www.opengalen.org/sources/publications.html. Major papers are additionally listed below

1996 The GALEN High Level Ontology (AL Rector, JE Rogers, P Pole) In: Fourteenth International Congress of the European Federation for Medical Informatics, MIE-96, Copenhagen, Denmark

1997 The GRAIL Concept Modelling Language for Medical Terminology (AL Rector, S Bechhofer, CA Goble, I Horrocks, WA Nowlan, WD Solomon) Artificial Intelligence in Medicine 9:139-171 (1997)

1997 Rubrics to Dissections to GRAIL to Classifications (JE Rogers, WD Solomon, AL Rector, P Pole, P Zanstra, E van der Haring) Fifteenth International Congress of the European Federation for Medical Informatics, MIE-97 Thessaloniki, Greece: 241-245

1997 Using the GRAIL language for Classification Management (PE Zanstra, EJ van der Haring, F Flier, JERogers, WD Solomon) Fifteenth International Congress of the European Federation for Medical Informatics, MIE-97 Thessaloniki, Greece: 441-445

1999 Reconciling Users Needs and Formal Requirements: Issues in developing a Re-Usable Ontology for Medicine (Rector, AL, Zanstra, P E, Solomon WD, Rogers JE, Baud, R, et al) IEEE Transactions on Information Technology in BioMedicine 2(4): 229-242

1999 A Reference Terminology for Drugs (Solomon WD, Wroe CJ, Rector, AL, Rogers JE, Fistein JL, Johnson P) Annual Fall Symposium of American Medical Informatics Association, Washington DC Hanley & Belfus Inc Philadelphia PA;:152-155

2001 GALEN Ten Years On: Tasks and Supporting tools (Rogers JE, Roberts A, Solomon WD, van der Haring E, Wroe CJ, Zanstra PE, Rector, AL) Proc MEDINFO2001 (London), Vimla Patel et al (Eds) IOS Press;: 256-260

2004 Development of a methodology and an ontological schema for medical terminology (Rogers JE) MD Doctoral Thesis, University of Manchester

2006 Ontological & Practical Issues in using a Description Logic to Represent Medical Concepts: Experience from GALEN (AL Rector, JE Rogers) School of Computer Science PrePrint, University of Manchester: CSPP-35:1-35

Appendix A: Translation Guide

The following pages show a small toy GRAIL ontology, containing examples of most of the principal GRAIL constructors, with each GRAIL statement (in bold) followed immediately by its OWL translation:

```
<!--TopCategory newSub [ClassA ClassB ClassC]. -->

<owl:Class rdf:about="#ClassA">
    <rdfs:subClassOf rdf:resource="#TopCategory"/>
</owl:Class>

<owl:Class rdf:about="#ClassB">
    <rdfs:subClassOf rdf:resource="#TopCategory"/>
</owl:Class>

<owl:Class rdf:about="#ClassC">
    <rdfs:subClassOf rdf:resource="#TopCategory"/>
</owl:Class>

<!-- Attribute newAttribute RoleA inverseRoleA allAll manyMany. -->
<owl:ObjectProperty rdf:about="#RoleA">
    <rdfs:label>RoleA</rdfs:label>
    <owl:inverseOf rdf:resource="#inverseRoleA"/>
    <rdfs:subPropertyOf rdf:resource="#Attribute"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#inverseRoleA">
    <rdfs:label>inverseRoleA</rdfs:label>
    <rdfs:subPropertyOf rdf:resource="#InverseAttribute"/>
</owl:ObjectProperty>

<!-- ClassA grammatically RoleA ClassB. -->
<owl:AnnotationProperty rdf:about="#G-SANCTION-RoleAAP"/>
<owl:Class rdf:about="#ClassA">
    <G-SANCTION-RoleAAP>ClassB</G-SANCTION-RoleAAP>
</owl:Class>

<!-- ClassA sensibly RoleA ClassB. -->
<owl:AnnotationProperty rdf:about="#S-SANCTION-RoleAAP"/>
<owl:Class rdf:about="#ClassA">
    <S-SANCTION-RoleAAP>ClassB</S-SANCTION-RoleAAP>
</owl:Class>
```

<!-- ClassA necessarily RoleA ClassB. -->

```
<owl:Class rdf:about="#ClassA">
	<rdfs:subClassOf>
		<owl:Restriction>
		<owl:onProperty rdf:resource="#RoleA"/>
			<owl:someValuesFrom rdf:resource="#ClassB"/>
		</owl:Restriction>
	</rdfs:subClassOf>
</owl:Class>

<owl:Class rdf:about="#ClassB">
	<rdfs:subClassOf>
		<owl:Restriction>
		<owl:onProperty rdf:resource="#inverseRoleA"/>
			<owl:someValuesFrom rdf:resource="#ClassA"/>
		</owl:Restriction>
	</rdfs:subClassOf>
</owl:Class>
```

<!-- ClassA extrinsically RoleA ClassB. -->

```
<owl:AnnotationProperty rdf:about="#RoleAAP"/>
<owl:Class rdf:about="#ClassA">
	<RoleAAP>ClassB</RoleAAP>
</owl:Class>
```

<!-- ClassA extrinsically RoleA ‘a piece of text’. -->

```
<owl:AnnotationProperty rdf:about="#RoleAAP"/>
<owl:Class rdf:about="#ClassA">
	<RoleAAP rdf:datatype="&xsd;string">a piece of text</RoleAAP>
</owl:Class>
```

<!-- ClassA extrinsically RoleA 2009. -->

```
<owl:AnnotationProperty rdf:about="#RoleAAP"/>
<owl:Class rdf:about="#ClassA">
	<RoleAAP rdf:datatype="&xsd;int">2009</RoleAAP>
</owl:Class>
```

<!-- ClassB grammaticallyAndSensibly RoleA ClassC. -->

```
<owl:AnnotationProperty rdf:about="#G-SANCTION-RoleAAP"/>
<owl:Class rdf:about="#ClassB">
	<G-SANCTION-RoleAAP>ClassC</G-SANCTION-RoleAAP>
</owl:Class>

<owl:AnnotationProperty rdf:about="#S-SANCTION-RoleAAP"/>
```

```xml
<owl:Class rdf:about="#ClassB">
	<S-SANCTION-RoleAAP>ClassC</S-SANCTION-RoleAAP>
</owl:Class>

<!-- (ClassA which RoleA (ClassB whichG RoleA ClassC) )name ClassD. -->
<owl:Class rdf:about="#ClassBwhichRoleAClassC">
	<owl:equivalentClass>
		<owl:Class>
			<owl:intersectionOf rdf:parseType="Collection">
				<rdf:Description rdf:about="#ClassB"/>
				<owl:Restriction>
					<owl:onProperty rdf:resource="#RoleA"/>
					<owl:someValuesFrom rdf:resource="#ClassC"/>
				</owl:Restriction>
			</owl:intersectionOf>
		</owl:Class>
	</owl:equivalentClass>
</owl:Class>

<owl:AnnotationProperty rdf:about="#SANCTION-LEVELAP"/><owl:Class
rdf:about="#ClassBwhichRoleAClassC">
	<SANCTION-LEVELAP rdf:datatype="&xsd;string">GRAMMATICAL</SANCTION-LEVELAP>
</owl:Class>

<owl:Class rdf:about="#ClassD">
	<owl:equivalentClass>
		<owl:Class>
			<owl:intersectionOf rdf:parseType="Collection">
				<rdf:Description rdf:about="#ClassA"/>
				<owl:Restriction>
					<owl:onProperty rdf:resource="#RoleA"/>
					<owl:someValuesFrom>
						<owl:Class rdf:about="#ClassBwhichRoleAClassC"/>
					</owl:someValuesFrom>
				</owl:Restriction>
			</owl:intersectionOf>
		</owl:Class>
	</owl:equivalentClass>
</owl:Class>
```