The GRAIL Concept Modelling Language for Medical Terminology

AL Rector, S Bechhofer, CA Goble, I Horrocks, WA Nowlan, WD Solomon
The GALEN Project
Medical Informatics Group, Department of Computer Science
University of Manchester, Manchester M13 9PL, England
Tel: +44-161-275-6133 FAX: +44-161-275-6932
Internet: galen@cs.man.ac.uk rector@cs.man.ac.uk
URL: http://www.cs.man.ac.uk/mig/galen/

This is an expanded version of paper which appeared in
***Artificial Intelligence in Medicine* 9:139-171 (1997)**

Abstract

The GALEN Representation and Integration Language (GRAIL) has been developed to support effective clinical user interfaces and extensible re-usable, models of medical terminology. It has been used successfully to develop the prototype GALEN Common Reference (CORE) Model for medical terminology and for a series of projects in clinical user interfaces within the GALEN and PEN&PAD projects. GRAIL is a Description Logic or Frame Language with novel features to support part-whole and other transitive relations and to support the GALEN modelling style aimed at re-use and application independence. GRAIL began as an experimental language. It has clarified many requirements for an effective knowledge representation language for clinical concepts. It still has numerous limitations despite its practical successes. The GRAIL experience is expected to form the basis for future languages which meet the same requirements but have greater expressiveness and more soundly based semantics. This paper provides a description and motivation for the GRAIL language and gives examples of the modelling paradigm which it supports.

Keywords: terminology, knowledge representation, description logic, concept models, re-use, medical records

Acknowledgements

This work supported in part by project number 2012 from the initiative on Advanced Informatics in Medicine (AIM) of the European Union, GALEN, and by project number GR/K42721, PAEPR, from the UK Engineering and Physical Sciences Research Council.

The GALEN Consortium consists of: University of Manchester (UK, Coordinator), Centre Hôpitalier Universitaire de Genève (Switzerland), Consiglio Nazionale delle Richerche (Italy), Conser Systemi Avanzati (Italy), the Association of Finnish Local Authorities, the Technical Research Centre of Finland (VTT) (Finland), GSF-Forschungszentrum, (Germany), Hewlett-Packard Ltd, University of Nijmegen (Holland), University of Linköping (Sweden), University of Liverpool (UK). The GRAIL language and software are copyright University of Manchester.

Table of Contents

1. Introduction

1.1 Compositional models of medical terminology

The GALEN Representation and Integration Language (GRAIL) has been developed to support effective clinical user interfaces and extensible re-usable models of medical terminology. It has been used successfully to develop the prototype GALEN Common Reference (CORE) Model for medical terminology [41] the GALEN Terminology Server [47], and for a series of projects in clinical user interfaces within the GALEN and PEN&PAD projects [2, 26, 27, 38, 39]. Commercial development of clinical user interfaces based on the GALEN models and GRAIL are in progress. GRAIL began as an experimental language. It is sufficiently well developed and documented that four independent implementations have been built, validated for conformity, and used for applications.

The overall goal of providing a rigorous foundation for medical terminology is shared with a number of other researchers, notable those of the CANON group [6, 12, 15, 20]. Other authors emphasise the role of shared terminologies — or more broadly shared 'ontologies' — in developing re-usable reasoning strategies for managing developing and maintaining computer-based support for clinical protocols [33, 34, 35, 51, 52]. A more general discussion of the goals for the GRAIL language can be found in [45] and of its use in the GALEN Terminology Server in [47]. Separate papers discuss the use of GRAIL for medical records [44] and its use in conjunction with natural language in [5]. The relationship of terminology to the Medical Logic Modules for exchange of clinical protocols can be found in [1]. A sketch of GALEN's model of anatomical concepts can be found in [43].

GALEN's knowledge representation language, GRAIL, is related to 'description logics' (sometimes called 'Frame languages') such as KL-ONE [10], CLASSIC [8], LOOM [31] and BACK [36] . A review of such systems can be found in [30]. It is also related to Conceptual Graphs [50], and to typed feature structures [13] . More precisely, GRAIL is a terminological language, analogous to the terminological component — or 'T-Box' [9] — of such systems. However, GRAIL has evolved separately since the late 1980s, driven by clinical applications and the need to provide a re-usable reference model for medical terminology. It has drawn both caveats and inspiration from Evan's work on representing concepts in MEDSORT [18, 19], Masarie *et al.*'s work on using frame systems as an interlingua for clinical terminologies [32], and Doyle and Patil's attempts to use an early terminological language (NIKL) to represent medical terminology [17]. As a result, GRAIL has taken different decisions from other description logics concerning which potential features to include in a 'terminological component' and how to approach the fundamental trade-offs in the design of a knowledge representation languages.

These choices have produced a system which has a distinct flavour and which has proved itself both as a representation for terminologies *per se* and as a basis for clinical applications, some now approaching commercial exploitation. More recent work on the foundations of description logics and their associated classification algorithms promises that future systems may avoid many of the restrictions and compromises imposed to make GRAIL practical while still retaining its key characteristics. However, the existing language has provided a basis for practical experiments which have set the requirements for future systems.

This paper describes the key features of GRAIL along with their motivation and use within the GALEN Common Reference (CORE) model of medical concepts. It presents examples of how GRAIL is used to solve a number of awkward modelling problems as well as the limitations and restrictions in the current version, some of which we hope to remove in future developments.

1.2 Motivation for GRAIL

The goal which has guided GRAIL's design is the development of a model of medical terminology which is:

- *Expressive* — Capable of representing the vast majority of the clinical terminology found in existing classifications as well as the greater detail required for medical records for clinical care. Overly narrow definitions of terminological reasoning such as those explored by Doyle and Patil [17] were therefore unacceptable.

- *Re-usable* — Suitable for formulating a unified formal model of medical terminology which could be used as an interlingua to represent existing and proposed controlled medical vocabularies for a variety of applications in a variety of specialties using a range of natural languages.
- *Extensible* — Able to be extended by users coherently to meet local requirements with a minimum need for reference to a central authority.
- *Generative and able to support direct data entry by clinicians* — Able to support structured data entry for all clinically significant information in the medical record [37, 46].
- *Computationally tractable and scalable* — Able to be scaled up to working systems which cover large areas of medicine while remaining efficient enough for use in practical clinical systems.

Achieving re-use of the taxonomies has been a particularly important motivating factor in the design of GRAIL. The two key strategies for achieving re-use supported by GRAIL are:

- *Coordinating independent taxonomies* — by forming composite entities and classifying them automatically. It is assumed in GRAIL models that most primitive entities will have exactly one primitive parent. All other subsumption is principled and performed automatically based on the descriptions and definitions of entities. For example the primitive parent of "stomach" might be "body part". If "stomach" were described as hollow, then it would automatically be classified under defined composite entity "hollow body part"; no further explicit subsumption by further primitives would be required.
- *Bridging levels of detail* — by having each entity represent, potentially, a range of concepts rather than a single concept. For example, "stomach ulcer", "ulcer of the wall of the stomach" and "ulcer of the mucosa of the wall of the stomach" might all be considered the same entity despite the different granularity of detail provided.

Key premises behind GRAIL's design include:

- Terminologies will continue to be developed separately from the databases and decision support systems which use them, just as the development of existing coding and classification systems such as the ICD and SNOMED has occurred separately from the development of existing clinical information systems. This means that the same mechanisms must be used to classify categories in the terminology and instances in the medical record using that terminololgy. Procedural attachments which act only on instances are therefore unacceptable.
- The representation must be executable and serve as the basis for a 'terminology server'.
- Most representations force application-specific decisions on implementors and that this is one major reason that re-use is elusive.
- Extensibility depends on transparent, rigorous structure.

1.3 Summary of distinguishing features of GRAIL

The result of these motivations is a language which shares many features with Description Logics and Conceptual Graphs but has a distinctive set of features and restrictions which we summarise below and explain in detail in the next section:

i) Extensions to classification:
   a) Support for 'essential criteria' or 'necessary statements' as well as defining criteria.
   b) Coordination of other transitive relations with subsumption using the 'specialised by' construct.
   c) Definition of irredundant canonical forms for each entity

ii) The use of multi-level sanctioning to constrain composition.

iii) A syntax which makes natural the formation of embedded definitions and which distinguishes clearly between defining and essential criteria.

iv) Homogeneous classification of categories (classes) and individuals (instances). There is no 'A-Box' in GRAIL in which statements affect the classification of individuals but not categories as there is in systems such as Loom[31].

v) Restrictions on negation, disjunction, equality, quantification and co-reference

1.4 Vocabulary and related systems

GRAIL	*Frame Language and Description Logics*	*Conceptual Graphs*	*CEN TC251*[1]	*Object Oriented Analysis*	*Smalltalk/*
entity	*frame/object*	*type/ individual*	*Semantic Category*	*object*	*object*
category	*class*	*type*	*Semantic Category*	*class*	*class*
individual	*instance/ individual*	*individual*	*individual (instance)*	*instance*	*instance*
attribute	*slot/role/ attribute*	*attribute*	*semantic link*	*relation/ attribute*	*method/ instance variable*
anonymous composite entity	-	*anonymous type*	-	-	-
named composite entity	*defined object*	*defined type*	*defined concept*	*aggregation/ Relation with immutable attribute types*	-

Table 1: Rough equivalencies of vocabulary between related systems. Note that the word"attribute" is used to mean a single-valued (functional) role in many description logics.

GRAIL shares much of this basic structure with frame languages, Conceptual Graphs and, roughly, with object oriented databases, languages such as Smalltalk or C^{++}, and with the conceptual framework being promulgated by CEN TC251 and its working Groups [14, 48]. However, the vocabulary used varies and can lead to confusion. GRAIL's vocabulary has been selected wherever possible to be distinct both from that of the Smalltalk programming language in which it was originally implemented and from the language used in the CEN TC251 and other standards bodies — at least as of the time the vocabulary was established. *Table* 1 gives a useful set of rough equivalencies in the vocabulary between GRAIL, Frame languages, Conceptual Graphs and the usage in Smalltalk and Object Oriented analysis.

2. Description of Major Features of GRAIL

2.1 Structure and Subsumption of GRAIL Entities and Comparison of Vocabulary.

2.1.1 Entities, statements and criteria

A GRAIL model consists of a network of nodes called 'entities' and directed arcs called 'statements'. Statements are labelled by special entities called 'attributes'; hence a statement can be considered as a triple of the form:

TopicEntity-Attribute-ValueEntity.

Every attribute has an inverse written *inv Attribute*, (or given a separate name) so that an inverse can be created for every statement.

ValueEntity-inv Attribute-TopicEntity.

1 See reference [45]

If both a statement and its inverse are present in the model, the statement is said to be 'bi-directional'; otherwise it is said to be 'uni-directional'. By convention, most attributes are written in the form *hasX* and their inverse in the form *isXOf, e.g. hasLocation* and its inverse *isLocationOf.*

For many purposes it is convenient to think of statements as consisting of a topic entity and a 'criterion' consisting of an attribute-value pair:

TopicEntity-Criterion

where

Criterion = Attribute-value.

Entities are divided into 'categories' and 'individuals'. Categories can be either elementary or defined.

The elementary categories and attributes are organised in a subsumption taxonomy which must be an acyclic directed graph. Subsumption is often written as *E1* ≥ *E2*, meaning "*E1* subsumes or is equal to *E2*". Attributes are likewise organised into a subsumption taxonomy which must be an acyclic directed graph. Definition of composite attributes is not supported.

2.1.2 Composite entities, definitions and essential criteria

Composite entities are created using the operator *which,* for example:

BodyPart which hasLaterality leftLaterality

Hand which hasLaterality leftLaterality

Each composite entity consists of a 'base' and a 'definition'. The base consists of an elementary entity, and the definition consists of a set of criteria.

In addition, both elementary and composite entities may be further described by 'essential criteria' which are necessary to the entity but do not form part of any sufficient set of criteria to recognise it. Essential criteria provide an important part of GRAIL's ability to bridge levels of detail and coordinate taxonomies. For example, being "an ulcer located in the stomach" is sufficient to recognise "gastric ulcer" but the essential criteria include that it is further located in the "mucosa of the wall of the stomach". The operations are defined so that both these descriptions and any other description with an intermediate level of detail — *e.g.* "ulcer of the wall of the stomach" — correspond to the same entity. Essential criteria extend the notion of 'terminological knowledge' to include restricted forms of information which must be represented assertionally in other formalisms.

All criteria are inherited indefeasibly along the subsumption taxonomy.

An entity in a model may therefore be considered as a four-tuple consisting of an elementary entity and three criteria sets:

<*base, definitional criteria, essential criteria, inherited criteria*>

The set of definitional criteria is often referred to simply as the 'definition'. GRAIL entities can be thought of as representing a region rather than a point in category space bounded at the top by the definition and at the bottom by the complete criteria set as shown in *Fig.* 1.

2.1.3 Criteria sets and formal subsumption

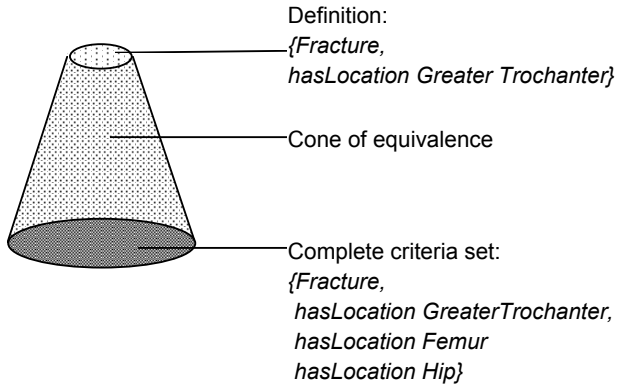

Figure 1: The cone of equivalence of a GRAIL entity.

One criterion subsumes another if both its attribute and value subsume the other's. (This definition is extended in Section 2.3 below.) One criteria set subsumes another if every criterion in the first set subsumes some criterion in the second. The generalised union (or 'meet') of two criteria sets consists of a set such that every criterion in either set subsumes some criteria in their union and no criterion in the union subsumes any other criterion in the union.

For any entity, the generalised union of the definition, the essential criteria and the inherited criteria is referred to as the 'complete criteria set' or 'complete description'. Two entities are considered equal if they have the same base and the same complete criteria set. One entity subsumes another if the bases subsume and the *definition* of the first entity subsumes the *complete criteria set* of the second.

The effect of the definition of subsumption is that each GRAIL entity may be thought of as representing a region or 'cone of equivalence' in concept space rather than a single concept. This is shown diagrammatically in *Fig.* 1. Any query using the operator *which* that falls within the cone of equivalence will return the same entity. The canonical form is the top of the cone of equivalence. The complete criteria set is the bottom. The range of granularity which can be bridged by the model is represented by the height of the cone connecting them.

2.1.4 Descriptive Statements and the coherence of entities

Descriptive statements add criteria to an entities essential criteria set. Descriptive statements about categories are usually known as 'necessary statements'; descriptive statements about individuals are known as 'facts'. Both elementary and composite categories may be described by necessary statements. Necessary statements use the keyword *necessarily*; facts use the keyword *really*. For example:

Femur necessarily isComponentOf Thigh

HollowObject necessarily defines Cavity.

John really isObservedToHave Diabetes

Like restrictions in frame languages, essential criteria introduced by necessary statements can be tightened — *i.e.* made more restrictive —as one moves down the subsumption taxonomy but cannot be relaxed.

Necessary statement correspond to a subset of quantified assertions in other terminology languages. However, unlike most other terminology languages, necessary statements are considered part of the intensional meaning of the category and are used in classification. Correspondingly, 'necessary statements' are restricted in their use of disjunction, existential quantification, and negation by comparison to general 'assertions' in first order logic, Conceptual Graphs or the 'A-Box' of Frame Languages such as LOOM.

2.1.5 Consistency of criteria sets and conflicting criteria

Roughly speaking, two criteria conflict if they have the same single-valued attribute and different values (see 'cardinality' in Section 2.2.3 below). More precisely, two criteria conflict if the attribute of one subsumes or is equal to the attribute of the other, the subsuming attribute is single-valued, and

one criterion does not subsume the other. A criteria set is consistent if it contains no conflicting criteria. Any entity with conflicting criteria in its complete criteria set — whether acquired by way of its definition, necessary statements, or inheritance — is said to be inconsistent. (GRAIL assumes that sibling entities are disjoint for this purpose. It cannot recognise the existence of a common subsumed entity in this situation, and in this respect its reasoning is incomplete. In practice, this has proved a useful simplification.)

2.1.6 Normal forms, irredundancy and the 'cone of equivalence'.

Entities are normally printed in a normalised or canonical form, and the *which* operator performs certain normalisations on expressions while classifying them. The resulting pseudo-operation to produce the normal or 'canonical' form of an expression has three steps.

i) Prior to classification, the expression is flattened leftwards: *i.e.* expressions of the form:

((...(B which C_1) which C_2) ...which C_n)

are converted to the form:

B which (C_1 C_2 ...C_n)

ii) The resulting criteria set is taken as the revised definition and combined with the base and classified iteratively.

iii) Following classification, the most general equivalent definition is chosen so as to remove redundancies. A definition contains redundancies if it is subsumed by an alternative definition for which the operator *which* returns the same entity, *e.g.*

Hand which <hasLaterality right isDivisionOf UpperExtremity>

is printed simply to:

Hand which hasLaterality right

because hands are necessarily part of upper extremity so the criterion *isDivisionOf UpperExtremity* is redundant.

The *specialisedBy* mechanism can interact with the removal of redundant criteria, so that given the statements in 2.3 plus:

GreaterTrochanter necessarily isDivisionOf Femur,

it follows that *hasLocation-Femur* is redundant in the criteria set <*hasLocation GreaterTrochanter hasLocation Femur*> so that:

Fracture which <hasLocation GreaterTrochanter hasLocation Femur>

is normalised to the canonical form:

Fracture which hasLocation GreaterTrochanter

In effect, the canonical form is the top of the 'cone of equivalence' defined in 2.1.2.

(Note that step iii is actually performed after classification. This prevents the discovery in certain circumstances than an entity is embedded in itself which could otherwise lead to cycles in a description logic supporting transitivity [3, 4]. In this respect GRAIL sacrifices completeness for decidability [28]. The problem has not been noted to occur in practice.)

2.1.7 Categories and individuals

Any category may be individuated by the command *newIndividual*, *e.g.*

Patient newIndividual John

Describing an individual causes it to be reclassified; describing a category causes it to be specialised. In this respect GRAIL is closer to Conceptual Graphs' view of types as lambda abstractions of individuals than to the view taken by formalisms from the KL-ONE family.

Most description logics in the KL-ONE family require the user to choose a level of specialisation arbitrarily for the division between class and instance — 'category' and 'individual' in GRAIL's vocabulary. As described dramatically by Brachman in [11], the choice of the level of specialisation at which to make the division depends on the application. In addition to limiting re-use, one of the earliest observations which led to GRAIL was that, in medicine, such arbitrary distinctions are often untenable even within a single application. For example, consider a drug database, should we choose

"aspirin", "children's aspirin", "children's flavoured aspirin", "children's flavoured aspirin by XYZ company", "children's flavoured aspirin by XYZ company from lot 1234" etc. as the level of specialisation for an individual? GRAIL side-steps this choice by representing only the aspirin in particular physical tablets as individuals. All other notions are represented as categories.

As the goal of GALEN was to separate of the categories used to represent terminology from the individuals used in the medical record to allow them to be developed separately, GALEN focuses on categories as does the remainder of this paper.

2.2 Constraining composition: Sanctioning, genericity and cardinality.

2.2.1 Multilevel sanctioning

Sanctioning statements constrain how categories may be used in the formation of composite entities but do not add to the category's description. Hence sanctions have no effect on how a category is classified. Sanctioning statements are analogous to canonical graphs in Conceptual Graph Theory. They are the converse of restriction as found in typical description logics of the KL-ONE family. In description logics, the metaphor is that everything is permitted and increasingly stringent 'restrictions' are applied as one goes from general to specific. In GRAIL and Conceptual Graphs, the metaphor is that nothing is permitted until it is first sanctioned. New sanctions can be added to as one goes down the taxonomy from general to specific, but sanctions cannot be narrowed There are three levels of sanctioning statements in GRAIL.

- *Conceivable* — statements that an attribute exists and may be used.
- *Grammatical* — statements that an abstraction is useful for querying but not sufficiently constrained for generation. Grammar level sanctions are roughly equivalent to function 'signatures' and are applied at a similar level to restrictions in other frame languages.
- *Sensible*[2] — statements that the constraints are sufficiently tight that all and only the compositions sanctioned can be generated.

Each level of sanctioning statement must be sanctioned by the one above — grammar-level sanctions by conceivable-level sanctions and sensible-level sanctions by grammar-level sanctions. Descriptive statements and criteria used in definitions of descriptive entities must be sanctioned at the sensible level; definitions of 'generalised' or 'query entities' need only be sanctioned at the grammar-level. If a statement is sanctioned at two or more different levels along different paths in the taxonomy, then the lower level sanction applies.

The distinction between grammar and sensible level sanctions allows both coarse-grained and fine-grained constraints to be defined. The distinction is useful because the constraints on queries are different from those on descriptions. For example the phrase "Lesion located in tissue" makes sense as a query even though it may produce an empty reply. However, not all lesions can sensibly be located in every tissue. Only some specialisations are sensible as statements about patients. Others — *e.g.* "Abscess located in blood" — are nonsense. This pattern is represented in GRAIL by sanctioning "Lesion located in tissue" at the grammatical-level and sanctioning "abscess located in solid tissue" at the sensible-level.

Correspondingly, the operator *which* has two variants:

which	requires the description to be sanctioned at the sensible level and is used for statements about individuals and generation;
whichG	requires the description to be sanctioned only at the grammatical level and is used for queries and abstract statements expressing constraints within the model.

2 The names for the levels, particularly the 'sensible' level have been hotly debated over time. What is now called the 'sensible' or 'sense' level was originally called 'possible', then 'plausible', but both caused confusion.

type	Level	Use	Example
Sanc-tion-ing	conceivable	Attribute declaration	*TopThing conceivably hasLocation TopThing.* *TopThing conceivable hasCause TopThing.* *TopThing conceivably isComponentOf TopThing.*
	grammatical	Queries and abstract statements	*Lesion grammatically hasLocation BodyPart.* *Lesion grammatically hasCause Process.* *Process grammatically haCause Process.* *BodyPart grammatically isComponentOf BodyPart*
	sensible	Generation and semantic validation	*Fracture sensibly hasLocation Bone.* *Fracture sensibly hasCause Osteoporosis.* *Osteoporosis sensibly hasCause PostMenopausalChange.* *Femur sensibly isDivisionOf Thigh*
De-script-ive	definition	Sufficient criteria for recognition	*Fracture which* < *hasLocation Femur* *hasCause (Osteoporosis which* *hasCause PostMenopausalChange)*>
	necessary statement	description of essential properties	*Femur necessarily isComponentOf Thigh.*

Figure 2a. Use and examples of levels of statement. (NB the statements using conceivably are inserted automatically when attributes are defined but are included here for completeness.)

Composite entities returned by *which* are known as 'descriptive entities'; composite entities returned by *whichG* are known as 'generalised entities' or 'query entities'. Therefore, to be strictly correct, the example from Section 2.1.2 should have been expressed as:

BodyPart whichG hasLaterality leftLaterality

Hand which hasLaterality leftLaterality

Since *hasLaterality* is only sanctioned at the grammatical level for *BodyPart* but is sanctioned at the sensible level for *Hand*.

2.2.2 Generation and the use of sanctions

The use and examples of the three levels of sanction plus definitions and necessary statements is illustrated in *Fig.* 2a in text form. The definitions are presented in diagrammatic form in *Fig.* 2b and the necessary statements in *Fig.* 2c. Each statement is sanctioned by the statement above. Every definition and necessary statement is sanctioned by a sense-level sanction. Sanctions at two levels may correspond and the compiler accepts abbreviations for such cases — *e.g.* in *Fig.* 2a we might have written: Femur *sensiblyAndNecessarily isDivisionOf Thigh.*

All classification of composite entities is automatic. Subsumption due to automatic classification is shown by dotted arrows in the diagrammatic representation. Note particularly how the necessary statement in causes *Femur* to be classified automatically as a *BodyPart whichG isComponentOf Thigh* as shown in *Fig.* 2c.

With only slight extension of the model shown in *Fig.* 2, we can generate all relevant combinations of fractures of the various parts of the right and left femur caused by trauma or osteoporosis, and by osteoporosis caused by post menopausal changes, parathyroid disease, cancer, etc. Furthermore we can create all of the potential abstractions under which they might be classified for different purposes as and when needed, *e.g. Lesion whichG isCausedby PostmenopausalChange.* While standard classifications would be unlikely to include such a category, specialised classifications for elderly care or gynaecology might require it. The decision of whether or not to create such an abstraction need not be made in advance but can be left to individual applications.

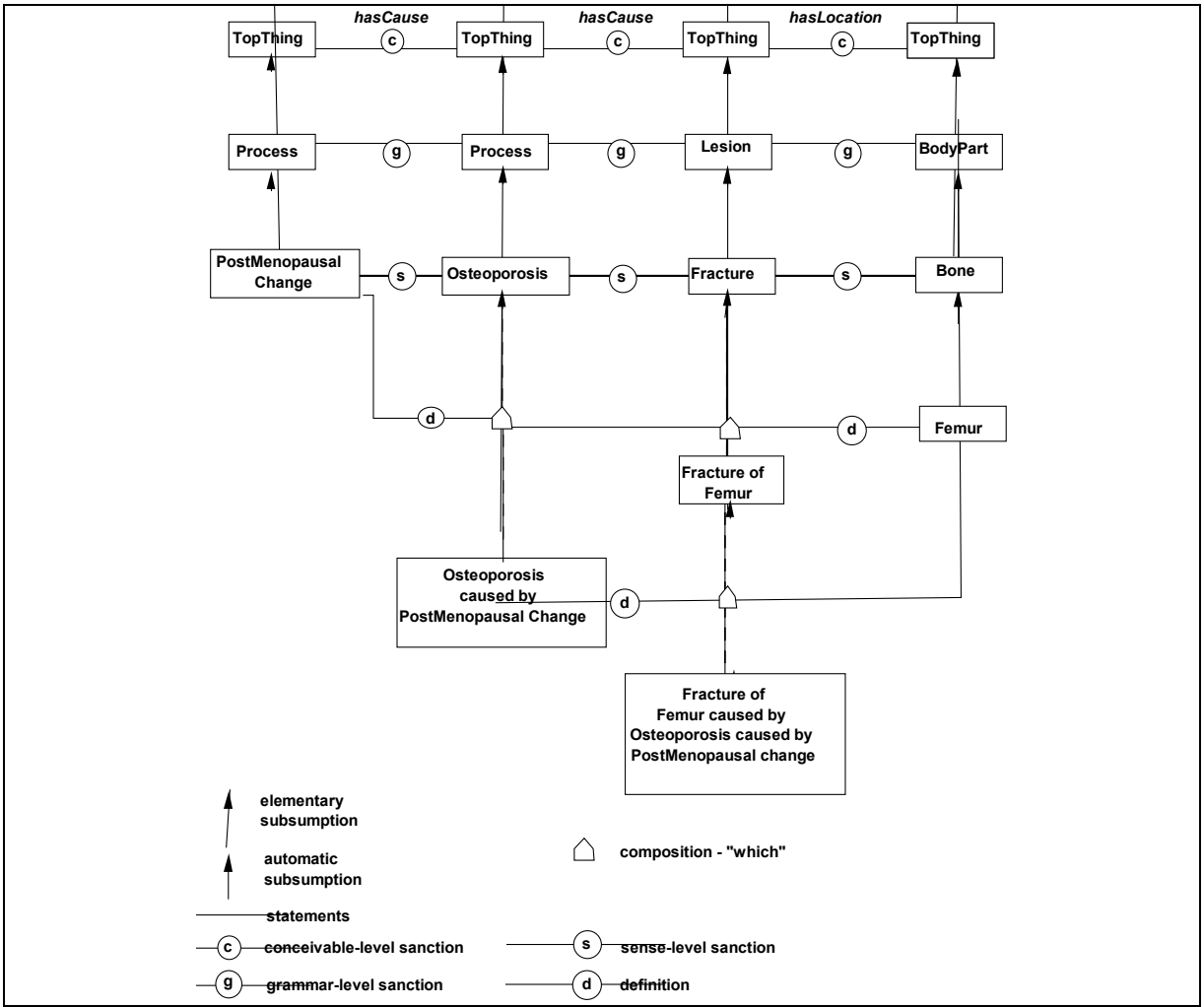

Figure 2b. Diagram of the definitions in *Fig*. 2a showing three levels of sanctions for each definition, each sanctioned by the level above. The symbol is equivalent to the operator *which*. Solid arrows indicate subsumptions entered explicitly by the modeller. Dotted arrows indicate subsumption provided automatically by the GRAIL classification mechanism. For clarity, cardinalities of statements have been omitted, and the names of entities have been given in their natural language rather form rather than GRAIL syntax.

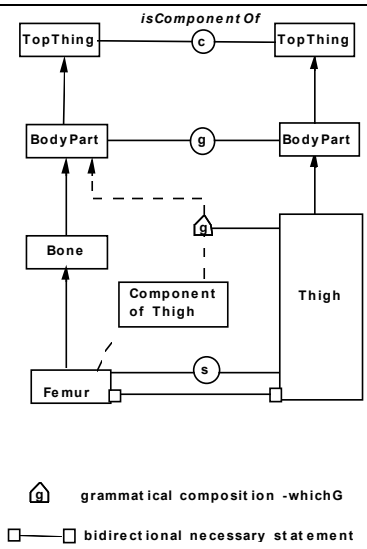

Figure 2c. Diagram of effect of necessary statements and automatic classification in *Fig*. 2a. "Femur" is automatically classified as a "component of the thigh" because of the necessary statement. Note also the use of *whichG*, diagrammed as which needs only to be sanctioned by a grammatical-level statement

2.2.3 Cardinality

All attributes have a cardinality, but the cardinalities in GRAIL are restricted to those commonly used in database schemata — one-one, one-many, many-one, and many-many. This restriction was chosen because interactions of more general cardinality constraints are known to be an important factor in the computational intractability of other description logics [36], and very few concepts were found in existing medical nomenclatures which required more complex cardinality constraints.

(Note that in the KL-ONE family of systems and some database communities single-valued links are called 'attributes', multi-valued links are called 'roles', and the word 'cardinality' is reserved for more complex numerical restrictions such as '>3' or '$\leq 2 \wedge \geq 0$') GRAIL's usage corresponds to that commonly used in object-oriented analysis and other parts of the database community.)

2.2.4 Interaction of Coherence and Sanctioning

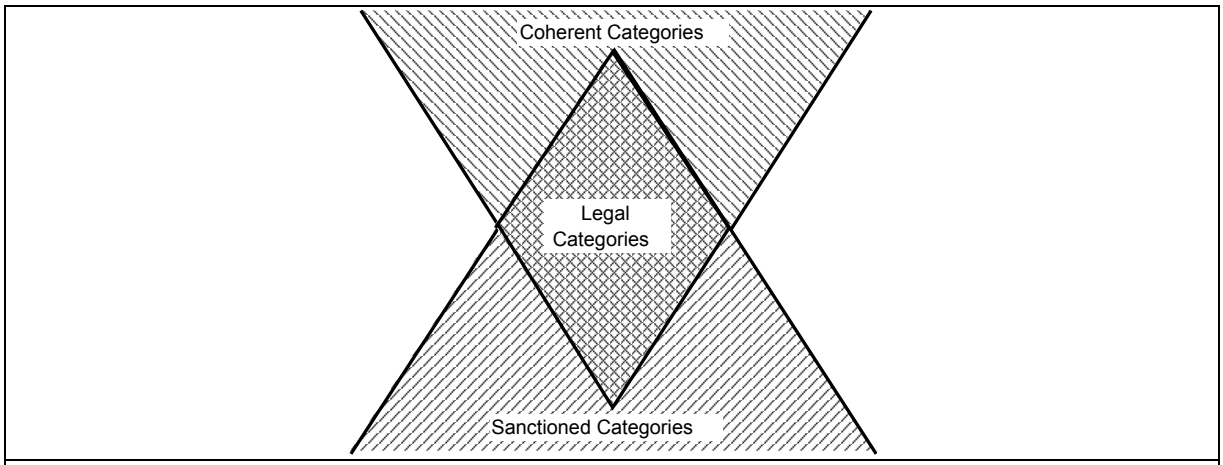

Figure 3: Interaction of constraints represented by coherence and sanctions.

Sanctioning and coherence are orthogonal. Sanctions can only be added and therefore become more lax as one moves down the taxonomy. Coherence is controlled by necessary statements which are indefeasible and therefore providing increasingly narrow constraints as one goes down the taxonomy. Legal categories are those which are both coherent and sanctioned as shown schematically by the intersection in *Fig.* 3.

2.3 Coordination of taxonomies

A key strategy in GRAIL for achieving application independence is to separate concepts into their components, classify the components separately, and then recombine them into composite entities and classify them on the basis of their definitions and descriptions. Many of the taxonomies to be coordinated depend on the subsumption relation — for example the taxonomy of processes and kinds of causative organisms. However, many do not.

Many medical concepts are the combination of an anatomy (topography) and a lesion or process — *e.g.* "ulcer of the stomach", "fracture of the femur" — or by a process and a cause — *e.g.* "viral hepatitis", "diabetic retinopathy". The organisation of anatomy is strongly dependent on part-whole relations and the organisation of causal structures by causal relations, both of which are transitive in many situations. Since all reasoning in GRAIL is by means of classification in the subsumption hierarchy, these other transitive relations must be coordinated with the primary subsumption hierarchy. The relationship is not straightforward: the "shaft of the femur" is a part of the femur rather than a kind of the femur, but a "fracture of the shaft of the femur" is a kind of a "fracture of the femur" and a "femur with a fractured shaft" is a kind of a "fractured femur".

The basic schemata are shown in *Fig.* 4a along with an example and the underlying semantics.

The same construct provides for transitive relationships as shown in *Fig.* 4b. For convenience, the construct *A* *specialisedBy* *A* can be abbreviated to *A* *transitiveUp* or to *inv* *A* *transitiveDown*.

Schemata:

A1 specialisedBy *A2*

implies

E1 which *A1 V* ≥ (1)
E1 which *A1 (E2* which *A2 V)*

and

V which *inv A1 E1* ≥ (2)
V which *inv A2 (E2* which *inv A1 E1)*

Example:

hasLocation specialisedBy *isDivisionOf*

implies

Fracture which *hasLocation Femur* ≥ (1)
Fracture which *hasLocation (AnatomicalNeck* which *isDivisionOf Femur)*

and

Femur which *isLocationOf Fracture* ≥ (2)
Femur which *hasDivision (AnatomicalNeck* which *isLocationOf Fracture)*

Semantics:

specialisedBy(A1, A2)

implies
$\forall x\ y\ z\ .\ A_1(x, y) \wedge A_2(y, z) \rightarrow A_1(x, z)$ (1)

from which it follows that by re-arrangement:
$\forall x\ y\ z\ .\ A_2^{-1}(z, y) \wedge A_1^{-1}(y, x) \rightarrow A_1^{-1}(z, x)$ (2)
[restriction: $A_1 = A_2$ or *specialisedBy*(A_2, A_2)]

Figure 4a: The two basic schemata for specialisedBy construct with an example and semantics. Numbers in parentheses indicate schema 1 or schema 2 for later reference.

Schema:

A specialisedBy *A*

implies

E1 which *A1-V1* ≥ (1)
E1 which *A1-(V2* which *A1-V1)*

Example:

isDivisionOf specialisedBy *isDivisionOf*

implies

Part which *isDivisionOf Whole* ≥ (1)
Part which *isDivisionOf (OtherPart* which *isDivisionOf Whole)*

Figure 4b: Construction of transitive relations using specialisedBy construct. Numbers in parentheses indicate the relevant schema.

Hand necessarily *isDivisionOf UpperExtremity*
Finger necessarily *isDivisionOf Hand*

implies

BodyPart which isDivisionOf UpperExtremity ≥ (1)
Finger

Figure 4c: Necessary statements affect classification. Numbers in parentheses indicate the relevant schema.

Necessary statements affect classification, so that given the necessary statements in *Fig.* 4c the classification shown follows automatically.

In order to accommodate the *specialisedBy* construct, the definition of subsumption of criteria must be generalised from that given in Section 2.1.3 so that one criterion subsumes another if either the attributes and values subsume each other respectively or there is a path from the subsumed criterion to the subsuming criterion indicated by the *specialisedBy* declarations.

Likewise, the rules for sanctioning are extended so that:

If A_1 *specialisedBy* $A_2 \wedge$
E_1 *which* A_1-E_2 is sanctioned,
then E_1 *which* A_1-(E_3 *which* A_2-$E2$) is sanctioned.

For example, in *Fig* 4a, the location of a fracture in the "anatomical neck of the femur" is automatically sanctioned by the sanction for a fracture in the "Femur" because

hasLocation specialisedBy isDivisionOf.

For convenience, particularly in diagrams, A_1 *specialisedBy* A_2 may also be written as A_1 *refinedAlong* A_2^{-1}.

The *specialisedBy* construct has a wide range of uses within GRAIL models. Further examples can be found in Sections 3.1 and 3.2.2.

2.4 Classification, Models and Names

2.4.1 Classification

Classification is the only form of inference in GRAIL. The primary operation on GRAIL is the operator *which* which classifies proposed entities on the basis of their definitions. All queries are performed using this operation. The 'answer' to the query is the taxonomy of entities subsumed by the result of the *which* operator. Intuitively the result is the simplest entity whose definition subsumes the expression and whose complete criteria set is subsumed by the expression. More formally, in the notation of 2.1.2:

B *which* Cs where $B = \langle BB, BD, BE, BI \rangle$ and Cs is a criteria set.

is the simplest entity

$E = \langle BB, D, E, I \rangle$ *such that* $D \geq (Cs \cup BD) \geq (D \cup E \cup I)$

Because of the presence of essential criteria, classification in GRAIL is iterative. An entity is first classified on the basis of its definition; its complete criteria set is then determined; it is then reclassified on the basis of any new criteria inherited; and the process is repeated until no new criteria are found. The restrictions in GRAIL ensure that the process is monotonic — *i.e.* that adding additional criteria during the classification process can never cause previously found criteria to be removed. This is a key property in guaranteeing that classification terminates, *i.e.* that it is decidable.

2.4.2 Bases, coherence and equivalence

For any GRAIL model, there is at least one 'basis' which consists of just those statements which are required to generate all of the entities in the extended model using *which*. Any GRAIL model is said to be an 'expansion' of its basis.[3] Two GRAIL models are said to be 'coherent' if a common basis can be found (up to renaming). Two GRAIL models are said to be equivalent if exactly the same entities (up to renaming) are returned from each using the operator *which*.

---

3 The use of "expansion" here is equivalent to what is often termed the "extension". However, the word "extension" has generated intense confusion when used in this context because of the conflict with usages such as "extensibility", "extension for cardiac diseases", and the confusion between the "extension of a category" and the "extension of a model or theory". The word "expansion" has therefore been adopted instead.

For practical purposes, the goal of 'reusability' is to achieve a common basis which is sufficient to generate the expansions suitable for a wide range of applications plus definitions of composite entities and interpretations which faithfully reflect the different usages in that range of applications.

2.4.3 Interpretations and naming

GRAIL separates definition and naming[4]. Any entity, composite or elementary can be named. For example:

(Hand which hasLaterality leftLaterality) name LeftHand.

(BodyPart which hasLaterality leftLaterality) name LeftSidedBodyPart.

An entity may have more than one name, but every name must refer to exactly one entity. Entities need not be named, and the availability of anonymous entities allows embedded expressions such as those in *Fig.* 2.

Naming an entity does not increase the expressive power of the model — *i.e.* it does not change how entities will be classified nor the results returned by the operator *which*. Hence naming does not affect either a model's coherence or equivalence with other models.

Strictly speaking, all names in a model belong to the 'primitive interpretation' of the model rather than to the model itself. However, since the purpose of any model is intimately intertwined with its interpretation, the model and its primitive interpretation are normally considered as a unit.

2.5 Extrinsic statements - linking to applications

Applications may use GRAIL models in two ways. They may simply ask questions of the terminology for their own processing. Alternatively, they may choose to use the terminology as a conceptual framework on which to hang information. Typically such information will be used in default-style reasoning. For convenience, GRAIL provides an operation to insert statements using the keyword *extrinsically* analogously to *sensibly*, *necessarily*, etc. Extrinsic statements are ignored by the classifier. GRAIL provides an operation to return the set of most specific extrinsic statements and their sources from a model but provides no further reasoning to decide amongst potential conflicts and ambiguities. Any disambiguation or other interpretation of extrinsic statements is left to the applications using them.

2.6 Restrictions and omitted features

The priorities in implementing the GRAIL Kernel have been determined by the existing corpus of controlled medical vocabularies and by the needs to keep the formalism computationally tractable. There are a number of restrictions and some features often found in related languages have been deliberately omitted.

- There is no operator for disjunction nor any operator for conjunction of elementary entities.
- Cardinality constraints are limited to one or many.
- Negation is not implemented in GRAIL-1.

In addition, The GRAIL Kernel is terminological, albeit the use of 'necessary statements' extends the meaning of terminological significantly. There is no default reasoning within the language itself, although the taxonomy makes an effective framework around which to implement default reasoners. Similarly there are no explicit quantifiers.

For practical reasons, there are also a number of other features not implemented in GRAIL-1.

- There are no shared variables; cyclical descriptions are not allowed and there is no SAME-AS operator.
- Normalisation in GRAIL-1 does not deal with cases such as the equivalence between:

  Hand which hasLaterality rightLaterality

  and

  Hand which isDivisionOf (Arm which hasLaterality rightLaterality).

---

4 Note that the phrases 'naming' and 'the unique name property' are used analogously to 'normal forms' in some of the description logic literature.

(This feature is referred to as 'exteriorization' in the GRAIL manuals)

- GRAIL-1 does not support symmetric attributes *i.e.* attribute such that *A=inv A e.g. AssociatedWith.*
- Statements and attributes cannot be themselves described.
- Cardinality is defined at the level of the attribute rather than the statement.
- Naming of composite attributes (Section 3.2.2 below) is not supported.
- Categories with multiple alternative definitions are not supported although the underlying semantics implies their existence, for example by saying that "Triangle necessarily has three sides" and "Three sided polygon necessarily has three angles". Entry of definitions or necessary statements which would make two entities equivalent are trapped as errors in the current implementation.
- The classification algorithm is not complete with respect to cycles introduced by transitive relations and a few other related situations. (Straightforward implementation of a complete algorithm leads to undecidability.)

3. Examples and Issues

3.1 Post-hoc classification

Consider the category defined in *Fig.* 2:

Fracture which <
  hasLocation
    (AnatomicalNeck which
      isDivisionOf Femur)
  hasCause (Osteoporosis which
      hasCause PostMenopausalChange)>

Then we can see that this category will be classified under many different abstractions, for example:

Fracture
Fracture which hasLocation LongBone.
Fracture which hasLocation (AnatomicalNeck which isDivisionOf LongBone).
Fracture which hasLocation Thigh.
Fracture which hasLocation Hip.
Lesion which isCausedBy Osteoporosis.
Lesion which isCausedBy PostmenopausalChange.
etc.

The choice of which abstractions to include in the system need not be made in advance. New abstractions can be created as required and the classifier will find the relevant items. Different applications can use different abstractions which will, in effect, organise the model along different axes. The classification can be built up *post hoc* from the definitions to fit requirements of each application rather being fixed *a priori*.

Extending the example somewhat further, a common operation to treat the condition might be expressed as:

Insertion which <
  hasIntendedOutcome (Fixation which
    actsOn (Fracture which <
        hasLocation (AnatomicalNeck which
          isDivisionOf Femur)
        hasCause (Osteoporosis which
          hasCause PostMenopausalChange)>
  actsOn (Pin which
    hasTargetLocation (AnatomicalNeck which
      isDivisionOf Femur))>

This category might be classified in a wide number of ways depending on the application ranging from "operations to repair fractures" to "Operations inserting devices" to "hip operations" to

"operations for complications of metabolic conditions". The entire expression or any of its components can be named to make the expression as compact as desired by particular users. Furthermore, as long as the underlying definitions are the same, two applications or two sites can use different naming conventions.

3.2 Modelling strategies and conventions

3.2.1 Coordination of Taxonomies: the use of roles, signs and symptoms

The prime strategy for achieving re-use in GRAIL models is to separate taxonomies [42]. The simplest example is the use of 'roles' to separate the function played by structures and substances from their form. For example, consider the notions of "vitamin", "hormone", "steroid" and "protein". Clearly, "vitamin" and "hormone" are functionally defined whereas "steroid" and "protein" are structurally defined. Roles are used in the GALEN CORE model to separate structures and substances from the role they play physiologically and to separate the means of observation from the thing observed. For example:

(ChemicalSubstance whichG playsPhysiologicalRole-VitaminRole) name Vitamin
(ChemicalSubstance whichG playsPhysiologicalRole-HormoneRole) name Hormone

In a similar way, the means of observation from the thing observed to distinguish between signs, symptoms, tests, etc., for example *Symptom* and *Sign* are defined by:

(Phenomenon whichG isShownBy PatientReport) name Symptom.
(Phenomenon whichG isShownBy ClinicalExamination) name Sign.

This use of 'roles' has become one of the fundamental mechanisms of the model. It has virtually replaced explicit statements that an entity has more than one parent. It is almost always preferable to represent the subsumption formally using a role as above, because the use of the role explains *why* the one entity is subsumed by the other whereas the use of an explicit subsumption link to two parents does not.

3.2.2 Bridging levels of detail

Different applications require different levels of detail — GRAIL seeks to avoid forcing a fixed choice of level of detail on developers. The GALEN CORE model seeks, on the one hand, to have a sufficiently detailed terminology for the most detailed of the applications supported and, on the other hand, to allow applications to ignore detail which is irrelevant to their specific needs. There are four methods for bridging levels of detail, (only the first two of which are fully implemented in GRAIL-1).

- *Named composite entities* allow applications to encapsulate complex descriptions in a single term.
- *Essential criteria* allow additional information to be added to an entity without affecting its definition. The normalisation mechanism guarantees that this information is invisible unless required. For example, for most purposes it is sufficient to consider "ulcers" to be located in the "stomach". However, in some applications, "ulcers" need to be classified under conditions of the "mucosa". In GRAIL if we add to the model the two statements:

  (Ulcer which hasLocation Stomach) necessarily hasLocation
  (Mucosa which isLayerOf Stomach)

  hasLocation specialisedBy isLayerOf

  then:

  Ulcer which *hasLocation (Mucosa which isLayerOf Stomach)* →
  Ulcer which hasLocation Stomach

  because the more specialised criterion *Mucosa which isLayerOf Stomach* adds nothing to the complete criteria set of *Ulcer which hasLocation Stomach* and is hence redundant. Mechanisms to allow more general specification of such constructs using a construct analogous to SAME-AS in KL-ONE will be included in GRAIL-1I.
- *Named composite attributes* allow applications, similarly, to deal only with a simplified view of the structure of the model, *e.g.*

  (hasFeature Severity hasState) namedAttribute hasSeverity

  Named composite attributes were initially considered unnecessary and were not supported in GRAIL-1. However, experience has shown that re-usability requires many attributes to be

expanded into attribute-entity-attribute sequences while many applications require simple attributes. Local user communities have strong views on which form to use. Accommodating local usage is more successful than attempting to impose a single solution.

- *'Interiorization'*, a special construct, hides awkward levels of embedding — *e.g.* it converts:

  (Lobe which isDivisionOf Lung) which hasLaterality right

  to:

  Lobe which isDivisionOf (Lung which hasLaterality right)

3.2.3 Use of iterative classification for simple inference

Consider the following. All hollow objects define spaces, but not all spaces are defined by hollow objects. Stating that an object is hollow should be sufficient to sanction the generation of the space defined by that object. Furthermore, stating additionally that the object defines a space is redundant. GRAIL represents these facts as follows. (The keyword *topicNecessarily* denotes a uni-directional necessary statement.)

(Structure whichG hasTopology hollow)
sensiblyAndTopicNecessarily definesSpace Space.

(Space whichG isDefinedBy
(Structure whichG hasTopology hollow))
name Cavity.

Stomach topicNecessarily hasTopology hollow.

From these statements it follows that *Stomach* is classified under *(Structure whichG hasTopology hollow)* and hence defines a *Space* named a *Cavity*.

However, were we to request the category *Stomach which definesSpace Cavity* the system would simply return *Stomach,* since the necessary statement has already added the criterion *definesSpace Space* to the description of *Stomach*, and since the definition of *Cavity* is simply that it is a *Space* defined by a hollow object.

Such simple one-stage or two-stage inferences are essential to capture the sense of much terminology and achieve intuitive behaviour from the system. However, they are difficult or impossible to capture with most description logics or with Conceptual Graphs without resort to the 'A-Box' which does not normally affect classification of categories. Use of such inference must be limited, however, to short chains or it will have an unacceptable impact on performance.

3.2.4 Levels of specification

Because GRAIL avoids the use of the class-instance boundary to indicate the critical level of specialisation or 'specification' for an application, the degree of specialisation of a category must be modelled explicitly in GRAIL rather than implied by the class-instance distinction. The primary anatomical concept of specification is the degree to which a given structure is unique in the body — the "liver", "right kidney", "left fourth finger" are each uniquely specified although the first is atomic, and the remainder are composites with increasingly complex patterns of selectors. Degree of specification is applied using necessary level statements: *e.g.*

(Finger which < hasOrdinalPosition OrdinalValueType
hasLaterality lateralityValueType>)
necessarily hasSpecificationLevel uniquelySpecified.

The anatomic level is intended to distinguish between the level which would usually be adequate in a retrieval of epidemiological statement — *e.g.* "injury to finger" — with the degree of specification required in a medical record or instruction to a surgeon — "injury to left fourth finger".

In addition, different applications require different measure of the degree of specification for different purposes. Each application can define analogous application specific levels of specification without compromising the re-usability of the model, *e.g.*

(Hepatitis which hasCause Virus)
necessarily hasApplicationSpecificationLevel
(adequatelySpecified which isWithRespectTo ApplicationSchema1)

3.3 General Issues

3.3.1 Dualities

Many medical phenomena come in pairs. The most common pairing is process and lesion, *e.g.* "ulceration" and "ulcer". Often the same word is used for both members of the pair, *e.g.* "erosion" (a process) and "erosion" (a lesion). Often, common usage blurs the distinction. For example we speak both of "gastritis" as if it were a process — "worsening", "progressive", "chronic", etc. — and of "gastritis" as something which can be seen through a gastroscope, has a diameter and a colour — clearly a physical lesion.

The phenomenon is quite general and not limited to pathological lesions, *e.g.* consider the process of "secretion" and the "secretion" which is its outcome. Language blurs these distinctions in some cases and emphasises them in others — *e.g.* there is a clear distinction in most languages between "viral hepatitis" and "hepatitis virus". A formal model must either mirror this pattern of blurring and distinguishing or provide formal transformations between the different forms. If it is to be independent of surface language, it must do so in a way which does not commit it to the usage in any one linguistic community. The CORE Model uses the attributes *hasSpecificOutcome, hasSpecificCause* etc. and makes an arbitrary, but consistent, choice to treat processes and organisms as primitive.

3.3.2 Interpretation of essential criteria and necessary statements

Statements such as *Hand necessarily isDivisionOf UpperExtremity* sometimes cause confusion. Questions are asked such as "What if the hand is severed." A key feature of GALEN's modelling style is to avoid such conundrums. What is being represented is terminology or "what can be said". Even if the hand has been severed, it is still reasonable to talk about it — *e.g.* to express the fact that the hand which is part of the left upper extremity is severed. Actual attachment must be expressed by a different attribute. The same pattern holds for all analogous cases.

3.3.3 External reasoners: Spatial and Temporal Reasoning

GRAIL supports reasoning for transitive relations and the *specialisedBy* construct. These can be used to provide a basic model for part-whole relations and some aspects of temporal inclusion. However, detailed spatial and temporal reasoning concerning contiguity, manipulation of temporal intervals and events, etc. is not part of the GRAIL language itself. The concepts required for such reasoning can usually be modelled in GRAIL, but their implications and interrelations are the province of specialised reasoners which use the terminology rather than of the terminological reasoning itself. GRAIL seeks to provide an application-independent framework for terminology which can be used by such application-specific reasoning systems.

3.3.4 Use as an interlingua: separation of application-specific and re-usable notions

The original idea of conversion between coding systems via an interlingua envisaged a single 'best' mapping. Experience has shown that the 'best' strategy for mapping depends on the application. For example, mappings for bibliographic searches generally should be as inclusive as possible Mappings for automatic encoding for standard returns almost always require an additional reasoning module to take the candidates and refine them. Mapping for some medical record purposes requires taking the nearest match based on a count of the number, and perhaps the priority of, the criteria contained in the various available codes. In each case the first step is to the standard GRAIL operation on extrinsic statements to find the set of most specific mappings to codes. Subsequent processing is application specific.

3.4 Special issues in medical applications

3.4.1 Mapping to existing nomenclatures and coding systems: *ClinicalSituations*, *presence* and *absence*, and "syndromes"

Pre-existing nomenclatures and coding systems often present special problems. The solution to two of these problems with respect to standard clinical coding systems further illustrates the features of GRAIL. The first problem concerns constructs which are common in the International Classification of Diseases (ICD) which contain expressions such as "with" and "without", for example ICD531.2 "Acute gastric ulcer with haemorrhage and perforation" [53] or SNOMED D5-32242 "Gastric ulcer with haemorrhage and perforation but without obstruction" [16]. These constructs present two difficulties:

- They refer to more than one concept;
- They involve negation.

The GALEN Core model solves the problem that ICD codes can map to more than a single concept by mapping all ICD codes to clinical situations rather than physiological structures or processes. The terms "with and without" are coped with by inserting "presence" or "absence" as states. Hence "Ulcer with haemorrhage but without perforation" becomes:

Clinical Situation which <
shows (presence which isStateOf Ulcer)
shows (presence which isStateOf Haemorrhage)
shows (absence which isStateOf Perforation)>

The above expression will be classified under any code showing the "ulcer", "presence of haemorrhage" or the "absence of perforation". Note that it is critical to use *PresenceOrAbsence which isStateOf Condition* rather than *Condition which hasState PresenceOrAbsence* because *Condition* ≥ *Condition which hasState absence* — for example that "ulcer" subsumes "absent ulcers" — which seems at least peculiar.

A further problem in existing nomenclatures is that clinicians treat the loose association represented by *ClinicalSituation* differently from how they treat standard 'syndromes' — associations of conditions believed to have a common, but often unknown, cause.

The construct which expresses the idea of 'syndrome' is shown below.

isStateOf specialisedBy includes

therefore:

presence which isStateOf Condition ≥
presence which isStateOf (Syndrome which includes Condition)

Hence, for example, consider a complex syndrome such as 'tetrology of falot' consisting of four associated conditions. This can be captured and related to an appropriate ICD code by:

Clinical Situation which shows (presence which
isStateOf (Syndrome which <
includes VentricularSeptalDefect
includes PulmonaryStenosis
includes RightVentricularHypertrophy
includes AorticOverriding>))

Using the first template from *Fig.* 2, it follows that the above category is subsumed by:

ClinicalSituation which shows (presence which isStateOf VentricularSeptalDefect)

Hence the code for "tetrology of falot" is subsumed under that for "ventricular septal defects".

3.4.2 What is a disease?

The medical notion of "disease" or "disorder" is difficult to capture in a strongly typed language such as GRAIL but essential to intuitive retrieval of medical concepts. "Diseases of the heart", "disorders of the kidney" or "disorders of circulation" are important, but heterogeneous concepts including lesions, processes, abnormal features and states.

To achieve this the GALEN CORE Model introduces two special abstractions:

- An entity which subsumes *Structure, Process, Feature* and *State,* currently named *Phenomenon*.
- An attribute which subsumes *isStateOf, isFeatureOf, hasLocation, isFunctionOf*, and *actsOn,* currently known as *isOf* or *hasLocativeAttribute*.

From these constructs we can then define *Disease* by:

(Phenomenon whichG hasPathologicalStatus pathological) name Disease.

DiseaseOfTheHeart then becomes simply:

Disease whichG isOf Heart.

Note that this still does not completely capture the common notion of "heart disease" found in existing classifications. For example "post myocardial syndrome" is caused by a cardiac changes

rather than located in the heart. We have so far been reluctant to elide causation and location, so that certain ICD chapter headings must map to two or more points in the GALEN taxonomy, one for cause and the other for location.

3.4.3 'Extrinsic knowledge': Why diagnostic criteria are not terminological

The goal of the GALEN Common reference model is to capture the 'terminological' part of clinical knowledge. The distinction is not easy to define. Technically, the boundaries are drawn at the limits of what can be represented within a set of constructs. Organisationally it is set by where consensus can be reached. Pragmatically, there is often a rough correspondence. For example the classification of "rheumatoid arthritis" as an inflammatory arthritis is uncontroversial. However, various different clinical groups have established various diagnostic criteria based on scoring systems — five out of seven set of seven criteria for one, six out of ten for another, etc. To require consensus on the diagnostic criteria before we can agree a terminology is fruitless. Coincidentally, though the individual criteria can be represented in GRAIL, notions such as "five out of seven" cannot be. Hence we take "inflammatory arthritis" as terminological and the diagnostic criteria as 'extrinsic' requiring a separate, possibly application specific reasoner. The overall hypothesis is that the two tests match well enough to be useful.

4. Discussion and Experience

4.1 Choice of features

GRAIL has been highly successful in supporting the primary strategy in this modelling style— 'coordinating taxonomies'. It overcomes many shortcomings of other languages in supporting the second key strategy — 'bridging levels of detail' — but it requires further enhancement before it can be regarded as fully successful in this regard. Overall it illustrates that there is a wider range of potential features to be included in terminological languages than might at first be thought from the literature.

The most successful novel features are:

- The extensions to classification in order to make classification of categories and individuals equivalent.
  - the *specialisedBy* mechanism for coordinating other relations with subsumption which has proved fruitful in a wide variety of contexts beyond it original intended use in coordinating partitive and generic relations. Other medical researchers have since proposed similar constructs [7, 49] and it is also closely related to the TRANSFERS-THRU construct from CycL [29].
  - The distinction between defining and essential criteria which has allowed levels of detail to be bridged to a much greater degree than had been expected and has dissolved numerous otherwise intractable arguments. GRAIL's extended notion of terminological reasoning allows most existing nomenclatures to be mapped to categories which are classified correctly without ad hoc procedural attachments or other opaque constructs.
- The substitution of an explicitly modelled 'level of specification' for the arbitrary distinction between category and individual (class and instance) which avoids an important class of arbitrary application-specific decisions in the model.
- The ease of forming embedded structures and anonymous types which greatly simplifies the modelling task.
- The limitation of GRAIL to a subset of inference and the demand that it be implementable. This excluded approaches such as that of KIF/Ontolingua tool kit which are designed as a non-executable means of describing and translating ontologies implemented in more limited executable formalisms [21, 24, 40]. Likewise, GRAIL models have no ambition to be complete theories of medicine, let alone of the world, such as those found in Cyc [25, 29].

The most troublesome omissions to be rectified in future versions are:

- The absence of any construct for defined attributes (defined relations) which makes it difficult to hide the complexity of the model from users. The need for such a construct is increasing as the model is being made more general to achieve re-usability.

- The absence of a construct analogous the SAME-AS operator in KL-ONE or to cyclical conceptual graphs which means that certain constructs can only be expressed imprecisely. Clinically plausible examples of such constructs are rare if the representation is confined to a single level of detail but become increasingly common as models attempt to bridge multiple levels of detail.
- The absence of negation although it has proved less of a problem than anticipated because of the development of the 'work arounds' in the modelling style such as the use of *presence* and *absence*. However, although the use of negation can be limited in formulating concepts, it is essential in expressing and querying entries in medical records. Implementing limited constructs for negation is now a priority.

Whether or not the limitations on cardinality are a serious problem remains controversial. There is no doubt that for some applications outside the initial target domain of clinical terminology, more sophisticated notions of cardinality and sets are essential. The number of situations in which they are required to represent notions in existing clinical terminologies remains surprisingly small but nonetheless a source of irritation to users.

The use of sanctions to constrain composition seems natural to those familiar with the use of Canonical Graphs in Sowa's Conceptual Graph Theory. The use of more than one level of canonical graph or sanctioning seems a modest extension. However, the use of sanctions for constraints is significantly different from the usage in the KL-ONE family of languages, and experience has shown that it can be surprisingly difficult to translate from one to the other [28]. The fundamental purpose of the sanctioning mechanism is to constrain genericity to support sophisticated user interfaces. Whether or not alternative approaches can be found and made efficient remains to be seen.

4.2 Satisfaction of Motivating Requirements

Returning to the initial motivations for developing GRAIL listed in Section 1.2:

- *Expressiveness* — GRAIL has proved sufficiently expressive to capture the content of existing medical terminologies with only rare exceptions. It supports modelling conventions capable of dealing with known awkward problems such as the aggregation of conditions in a single code in existing classifications. In general, it has been possible to work around the most irksome limitations such as the absence of negation. The model now covers almost all of the gastro-intestinal and cardiology sections of SNOMED and ICD-9 plus a wide range of common conditions, signs, symptoms, and associated disorders and is proving sufficient for a commercial data entry system for common conditions in general practice.
- *Re-usability and application independence* — Full demonstration of re-usability must await a wider range of applications which use the GALEN CORE Model. However, GRAIL has been successful in overcoming a number of identified barriers to re-use in other formalisms, and the same high level schemata support several different clinical data entry systems and the description of several different existing coding systems.
- *Extensibility* — The ability to name and classify detailed concepts from a restricted GRAIL basis forms a powerful method of extension. The strong constraints in the language also mean that many aspects of modelling style can be enforced so that new detailed concepts can be introduced with confidence.
- *Genericity and ability to support direct data entry by clinicians* — Support for direct data entry and generation is GRAIL's major success. User interfaces based on GRAIL have been implemented in several areas and are now being developed commercially. Automatic generation as an aid to formulation of classification systems has been demonstrated in prototypes and will be put to practical test on a significant scale in the next phase of the project.
- *Computational tractability and scaling* — Empirical experimentation indicates that classification time in the current GRAIL implementation is approximately linear with the size of the model [28]. The current GALEN CORE model contains on the order of seven thousand concepts (and the basis implies the possibility of generating indefinitely many more). Performance in the original Smalltalk implementation is now becoming marginal, but performance in the C^{++} version (roughly 30 times faster) suggests that models of five times this size or more are practical. In addition, empirical testing indicates that the performance is disproportionately

affected by the brute force implementation of a few unanticipated constructs such as *ClinicalSituation* which it should be possible to optimise in future versions.

4.3 Conclusion

GRAIL was developed because no system was available which provided the combination of features required and because experience elsewhere with existing systems had confirmed the unsuitability of those systems [17]. Most acutely, no system dealt with the interaction of partitive and other transitive relations with subsumption. The fact that the initial combination of features in its predecessor language, SMK [46], proved effective and tractable was gratifying and even surprising. The language has developed since to support a style of conceptual modelling which is proving effective in dealing with medical terminology and, increasingly, broader applications. The basic strategies of coordinating taxonomies and bridging levels of detail appear widely applicable.

Independently, over the last decade there have been major advances in description logics and classification algorithms. A number of the restrictions in GRAIL may now be unnecessary, and more powerful algorithms with provable results on completeness and decidability now exist for many more combinations of features. Experience with GRAIL has allowed us to make our requirements for description languages for medical terminology much clearer. Experiments in non-medical domains convince us that these requirements are relevant more widely than just medicine [22, 23]. A key challenge is to incorporate and extend advances made elsewhere in order to provide a successor language which is more expressive and soundly based while retaining the features which have made GRAIL successful.

References

[1] H. Ahlfeldt, N. Shashavar, X. Gao, K. arkad, B. Johansson, O. Wigertz, Data driven medical decision support based on Arden Syntax within the HELIOS environment, Computer Methods and Programmes in Biomedicine 45 (1994) S97-S107.

[2] L. Alpay, R. Baud, R. A-M, J. Wagner, C. Lovis, J.-R. Scherer, Artificial Intelligence in Medicine in Europe (AIME-93) Munich (1993)

[3] F. Baader, Augmenting concept languages by transitive closure of roles: An alternative to terminological cycles, International Joint Conference on Artificial Intelligence (IJCAI-91) (Morgan Kaufmann, 1991) 446-451.

[4] F. Baader, Augmenting concept languages by transitive closure of roles: An alternative to terminological cycles., Technical Report Deutsches Forschungszentrum für Künstliche Intelligenz GmbH (1990).

[5] R. Baud, C. Lovis, L. Alpay, A.-M. Rassinoux, J.-R. Scherrer, A. Nowlan, A. Rector, Modelling for Natural Language Understanding, C. Safrans, eds., Seventeenth Annual Symposium on Computer Applications in Medical Care (SCAMC-93) (McGraw Hill, Washington DC, 1993) 289-93.

[6] D. S. Bell, E. Pattison-Gordon, R. A. Greenes, Experiments in concept modeling for radiographic image reports, Journal of the American Medical Informatics Association 1 (1994) 249-262.

[7] J. Bernauer, H. Goldberg, Compositional classification based on conceptual graphs, Artificial Intelligence in Medicine Europe (AIME-93) Munich (1993)

[8] A. Borgida, R. J. Brachman, D. L. McGuinness, L. A. Resnick, CLASSIC: A Structural Data Model for Objects, ACM SIGMOD International Conference on Management of Data (ACM, 1989) 58-67.

[9] R. Brachman, R. Fikes, H. Levesque, An essential hybrid reasoning system; knowledge and symbol level accounts of KRYPTON, International Joint Conference on Artificial Intelligence (IJCAI-85) (Morgan Kaufmann, 1985) 532-539.

[10] R. Brachman, S. JB, An overview of the KL-ONE knowledge representation system, Cognitive Science 9 (1985) 171-216.

[11] R. J. Brachman, D. L. McGuinness, P. F. Patel-Schneider, L. A. Resnick, A. Borgida, Living with Classic: When and how to use a KL-ONE-like language, in Principles of Semantic Networks: Explorations in the representation of knowledge J. Sowa, eds. (Morgan Kaufmann, San Mateo, CA, 1991) 401-456.

[12] K. E. Campbell, A. K. Das, M. A. Musen, A logical foundation for representation of clinical data, JAMIA 1 (1994) 218-232.

[13] B. Carpenter, The Logic of Typed Feature Structures, Cambridge Tracts in Theoretical Computer Science (Cambridge University, Press, Cambridge, England, 1992).

[14] CEN TC251/WG2/PT02S, Model of Surgical Procedures, Technical Report CEN TC251 WG2 (1995).

[15] J. C. Cimino, G. Hripscak, S. Johnson, Knowledge-based approaches to the maintenance of a large controlled medical terminology., Journal of the American Medical Informatics Association 1 (1994) 35-50.

[16] R. Côté, D. Rothwell, SNOMED-International (College of American Pathologists, Chicago, 1993).

[17] J. Doyle, R. Patil, Two theses of knowledge representation: Language restrictions, taxonomic classification and the utility of representation services, Artificial Intelligence 48 (1991) 261-297.

[18] D. Evans, Final Report on the MedSORT-II Project: Developing and managing medical Thesauri., Technical Report Carnegie Mellon University, 1987.

[19] D. Evans, Pragmatically-structured, lexical-semantic knowledge bases for unified medical language systems., R. Greeness, eds., Proceedings of the Twelfth Annual Symposium on Computer Applications in Medical Care (IEEE Computer Society Press, Washington DC, 1988) 169-173.

[20] D. A. Evans, J. Cimino, W. R. Hersh, S. M. Huff, D. S. Bell, The Canon Group, Position Statement: Towards a Medical Concept Representation Language, Journal of the American Medical Informatics Association 1 (1994) 207-217.

[21] R. Fikes, M. Cutkosky, T. Gruber, j. V. Baalen, Knowledge Sharing Technology -- Project Overview, Technical Report Stanford University, Knowledge Sharing Laboratory, 1991.

[22] C. Goble, A. Rector, W. Nowlan, A. Glowinski, Conceptual, semantic and information models for medicine, in Information Modelling and knowledge Bases V. H. Kangassalo, H. Jaakkola, S. Ohsuga, B. Wangler, eds. (IOS Press, Amsterdam, 1994) 257-286.

[23] C. Goble, C. Haw, S. Bechhofer, Describing and classifying multimedia using the description logic GRAIL, Proceedings of the SPIE Conference on Storage and Retrieval of Still Image and Vidio IV, SPIE vol 2670 , San Jose (1995)

[24] T. R. Gruber, Toward principles for the design of ontologies used for knowledge sharing, Technical Report Stanford, Knowledge Systems Laboratory, 1993.

[25] R. Guha, D. Lenat, Enabling agents to work together, Communications of the ACM 37 (1994) 127-142.

[26] H. Heathfield, J. Kirby, Designing a patient record system for elderly care, 17th annual Symposium on Computer Applications in Medical Care, *SCAMC 93* (McGraw Hill, Washington, USA, 1993) 129-135.

[27] B. Horan, et al., Supporting a Humanly Impossible Task: The Clinical Human-Computer Environment, D. Diapers, eds., Interact 90 (Elsevier Science Publishers, B.V.North-Holland, 1990) 247-252.

[28] I. Horrocks, Masters thesis, University of Manchester (1995).

[29] D. B. Lenat, R. V. Guha, Building Large Knowledge-Based Systems: Representation and inference in the Cyc Project (Addison-Wesley, Reading, MA, 1989).

[30] R. MacGregor, The evolving technology of classification-based knowledge representation systems, in Principles of Semantic Networks: Explorations in the representation of knowledge. J. Sowa, eds. (Morgan Kaufmann, San Mateo, CA, 1991) 385-400.

[31] R. MacGregor, Inside the LOOM description classifier, SIGART Bulletin 2 (1991) 88-92.

[32] F. Masarie, R. Miller, O. Bouhaddou, N. Giuse, H. Warner, An interlingua for electronic interchange of medical information: using frames to map between clinical vocabularies, Computers in Biomedical Research 24 (1991) 379-400.

[33] M. Musen, Dimensions of knowledge sharing and reuse, Computers and Biomedical Research 25 (1992) 435-467.

[34] M. Musen, A. Schreiber, Architectures for intelligent systems based on reusable components, Artificial Intelligence in Medicine (1995) .

[35] M. Musen, S. Tu, A. Das, Y. Shahar, A component-based architecture for automation of protocol-directed therapy, P. Barahona, M. Stefanelli, J. Wyatts, eds., Fifth conference on Artificial Intelligence in Medicine Europe (AIME '95) (Springer, Pavia, Italy, 1995) 3-16.

[36] B. Nebel, Computational Complexity of Terminological Reasoning in Back, Artificial Intelligence 34 (1988) 371-383.

[37] W. Nowlan, A. Rector, Medical Knowledge Representation and Predictive Data Entry, M. Stefanelli, A. Hasman, M. Fiesch, J. Talmons, eds., Artificial Intelligence in Medicine Europe (AIME-91) (Springer-Verlag, Maastricht, 1991) 105-116.

[38] W. Nowlan, A. Rector, S. Kay, B. Horan, A. Wilson, A Patient Care Workstation Based on a User Centred Design and a Formal Theory of Medical Terminology: PEN&PAD and the SMK Formalism, P. Claytons, eds., Fifteenth Annual Symposium on Computer Applications in Medical Care. SCAMC-91 (McGraw- Hill, Inc, Washington DC, 1991) 855-857.

[39] W. A. Nowlan, Clinical workstation: Identifying clinical requirements and understanding clinical information, International Journal of Bio-Medical Computing 34 (1994) 85-94.

[40] R. S. Patil, R. E. Fikes, P. F. Patel-Schneider, D. McKay, T. Finin, T. Gruber, R. Neches, The DARPA Knowledge Sharing Effort: Progress Report, Principles of Knowledge Representation and Reasoning, Third International Conference (Morgan Kaufmann, Cambridge MA, 1992).

[41] A. Rector, Compositional models of medical concepts: towards re-usable application-independent medical terminologies, in Knowledge and Decisions in Health Telematics P. Barahona, J. Christensen, eds. (IOS Press, 1994) 133-142.

[42] A. Rector, Coordinating taxonomies: Key to re-usable concept representations, P. Barahona, M. Stefanelli, J. Wyatts, eds., Fifth conference on Artificial Intelligence in Medicine Europe (AIME '95) (Springer, Pavia, Italy, 1995) 17-28.

[43] A. Rector, A. Gangemi, E. Galeazzi, A. Glowinski, A. Rossi-Mori, The GALEN CORE Model Schemata for Anatomy: Towards a re-usable application-independent model of medical concepts, P. Barahona, M. Veloso, J. Bryants, eds., Twelfth International Congress of the European Federation for Medical Informatics, MIE-94 , Lisbon, Portugal, 1994) 229-233.

[44] A. Rector, A. Glowinski, W. Nowlan, A. Rossi-Mori, Medical concept models and medical records: An approach based on GALEN and PEN&PAD, Journal of the American Medical Informatics Association 2 (1995) 19-35.

[45] A. Rector, W. Nowlan, A. Glowinski, Goals for Concept Representation in the GALEN project, 17th Annual Symposium on Computer Applications in Medical Care (SCAMC-93) (McGraw Hill, 1993) 414-418.

[46] A. Rector, W. Nowlan, S. Kay, Unifying Medical Information using an Architecture Based on Descriptions, R. Millers, eds., 4th Annual Symposium on Computer Applications in Medical Care , SCAMC-90, (IEEE Computer Society Press, Washington DC, 1990) 190- 194.

[47] A. Rector, W. Solomon, W. Nowlan, T. Rush, A Terminology Server for Medical Language and Medical Information Systems, Methods of Information in Medicine 34 (1995) 147-157.

[48] A. Rossi-Mori, CEN Project Team PT003, Model for representation of semantics in Medicine, Technical Report CEN TC251 WG1,1995.

[49] J. Smart, M. Roux, Medical knowledge representation for medical report analysis, P. Barahona, M. Stefanelli, J. Wyatts, eds., Fifth conference on Artificial Intelligence in Medicine Europe (AIME '95) (Springer, Pavia, Italy, 1995) 53-66.

[50] J. Sowa, Conceptual Structures: Knowledge Representation in Mind and Machine (John Wiley & Sons, New York, 1985).

[51] S. Tu, H. Eriksson, J. Gennari, Y. Shahar, M. Musen, Ontology-based configuration of problem-solving methods and generation of knowledge-acquisition tools - application of PROTEGE-II to protocol-based decision-support, Artificial Intelligence in Medicine 7 (1995) 257-289.

[52] G. Vanheijst, S. Falasconi, A. Abuhanna, G. Schreiber, M. Stefanelli, A case-study in ontology library construction, Artificial Intelligence in Medicine 7 (1995) 227-255.

[53] World Health Organisation, Manual of the International Statistical Classification of Diseases, Injuries and Causes of Death (ninth revision) (World Health Organisation, Geneva, 1980).

Appendix 1: Brief summary of essential GRAIL operations and syntax

For complete details see reference manual and tutorial available at URL
http://www.cs.man.ac.uk/mig/galen/

Category *newSub* *string* *e.g.* *BodyPart* *newSub* *Stomach*	Creates a new elementary category with the name *string* subsumed by *Category*
$Category_1$ *addSuper* $Category_2$	Causes $Category_1$ to subsume $Category_2$. Both $Category_1$ and $Category_2$ must be elementary.
Attribute *newAttribute* *String1* *String2* *cardinality*	Creates a new attribute with name *string1* and inverse named *string2* with cardinality *cardinality*. *cardinality* is one *oneOne, oneMany, manyOne* or *manyMany*.
Category *necessarily* *Attribute* *Value* *e.g.* *Hand* *necessarily* *isPartOf* *UpperExtremity*	Enters a bidirectional necessary statement thereby adding the criterion *Attribute-Value* to the essential criteria set of *Category* and *inv* *Attr-Category* to the essential criteria set of *Value*
Category *topicNecessarily* *Attribute* *Value* *e.g.* *HollowObject* *topicNecessarily* *defines* *Cavity*	Enters a unidirectional necessary statement thereby adding the criterion *Attribute-Value* to the essential criteria set of *Category*.
Category *compositeOp* *Attribute* *Value* *e.g.* *Hand* *sensiblyAndNecessarily* *isDivisionOf* *Upper extgremity*	Sanctioning operators and *necessarily* may be combined into single operations for convenience. The meaning of such composite statements is the same as if the statements had been given individually.
Category *sanctioningOp* *Attribute* *Value* *e.g.* *Extremity* *sensibly* *hasLaterality* *Laterality*	Sanctions the criterion *Attribute-Value* for *Category* and the criterion *inv* *Attribute-Category* for *Value* at the level indicated by *sanctioningOp*. *SanctioningOps* are *conceivably*, *grammatically* and *sensibly*.
Base *whichOp* *CriteriaSet* *e.g.* *Hand* *which* *hasLaterality* *rightLaterality*	Returns the category defined by *Base* and *Criteria* set if possible. Base is a category. *whichOp* can be *which* or *whichG*. CriteriaSet can be of the form $A\ V$ or $<A_1\ V_1\ A_2\ V_2 ... A_n\ V_n>$ where the *A*s are attributes and the *V*s are entities. If the resulting category does not exist in the Model, it is created and classified. If the resulting category is coherent and sanctioned at the level indicated by *whichOp*, it is returned. If not, an error is returned and the category is discarded.
Attribute *transitiveDown*	Declares *Attribute* to be transitive downwards as defined in Section 2.3.
$Attribute_1$ *specialisedBy* $Attribute_2$	Declares $Attribute_1$ to be specialised by $attribute_2$ as defined in Section 2.3

Appendix 2: Brief Summary of Operational Semantics

The following is a summary of the operational semantics of GRAIL, restricted to what is currently implemented, which roughly follows the presentation in the paper. Individuals are omitted as categories are the main focus of GALEN's use of GRAIL. Although entities are described as four-tuples in the text, they are here condensed to three-tuples with the base indicated by the special pseudo-attribute *is-P* where *P* is a primitive. For brevity, minor variants such as the distinction between operators *which* and *whichG* have been omitted as has the convenient abbreviation which allows primitives to be introduced as children of composites. (Note that because GRAIL includes transitivitity, a strict first-order model theoretic semantics is not possible [3, 4]. Alternative semantics based on finite automata have been suggested but are beyond the scope of this paper. Similarly, two alternative model theoretic approaches to the semantics of multi-level sanctioning have been suggested.)

Model

A model consists of a tuple of sets as defined below:
<*Primitives, Attributes, sp, sa, Entities, Statements, inverse, cardinality, specialisedBy*> where

Primitives is a set of primitive symbols

Attributes is a set of primitive symbols disjoint from *Primitives*

sp is a relation on *Primitives* forming an acyclic directive graph.

sa is a relation on *Attributes* forming an acyclic directed graph

Entities is a set of entities as defined below constructed from the members of*Primitives* and *Attributes*.

Statememts is a set of statements as defined below constructed from the members of *Primtiives* and *Attributes*

inverse is a function on *Attributes* such that
$\forall A \in Attributes\ .\ inverse(inverse(A)) = A \wedge inverse(A) \neq A$

cardinality is a function from *Attributes* to *{many one}*

specialisedBy is a function from *Attributes* to *Attributes**

Model independent Primitives

The set *Qualifiers= {conceivable, grammatical, sensible, necessary}* and the ordering *conceivable>grammatical>sensible>necessary.*

The set of cardinality symbols *{one many}*

Structures In a model M=<*Ps As rps ras Es Ss inv card sb*> the following structures are defined:

Criteria	Ordered pairs of the form *A-E* or *is-P* where $A \in As$, $E \in Es$, $P \in Primititives$
Entities	Triples of the form <*D, N, I*> where *D,N,I* are all sets of criteria and there is exactly one criterion in *D* of the form *is-P* where *P* is a primitive
Complete criteria set of an Entity	$ccs(<D,N,I>) = D \cup N \cup I$
Elementary Entities	Elementary entities are constructed from primitives by the function *ee* derived from the lemma that: $\forall p \in Ps \; . \; \exists_1 E \in Es \; . \; E=<\{is\text{-}P\}, N, I>$ written *ee(P)* The uniqueness of elementary entities can be proven from the invariants but is beyond the scope of this presentation.
Statements and inverses	Four-tuples of the form $<T\ A\ V\ Q>$ where $T,V \in Es$, $A \in As$, ,$Q \in .Qualifiers$ $inverse(<T\ A\ V\ Q>) = <V\ inverse(A)\ T\ Q>$

Subsumption In a model M=<*Ps As rps ras Es Ss inv card sb*> subsumption is defined by:

Entities	$E_1 \geq E_2$ iff $E_1 = <D_1, N_1, I_1> \wedge D_1 \geq ccs(E_2)$
Criteria Sets	$Cs_1 \geq Cs_2$ iff $\forall c_1 \in Cs_1 \; . \; \exists c_2 \in Cs_2 \; . \; c_1 \geq c_2$
Criteria	$A_1\text{-}V_1 \geq A_2\text{-}V_2$ iff one of the following holds: a) $A_1\text{-}V_1 = is\text{-}P_1 \wedge A_2\text{-}V_2 = is\text{-}P_2 \wedge P_1 \geq P_2$. b) $A_1 \geq A_2 \wedge V_1 \geq V_2$ c) $\exists A_3\text{-}V_3 \in ccs(V_2) \; . \; A_2 \in specialisedBy(A_3) \wedge V_1 \geq V_3 \wedge A_1 \geq A_3$ d) $\exists A_3\text{-}V_3 \in ccs(V_2) \; . \; inv\ erse(A_3) \in specialisedBy(inverse(A_2)) \wedge V_1 \geq V_3 \wedge A_1 \geq A_3$ e) $\exists A_3\text{-}V_3 \; . \; A_1\text{-}V_1 \geq A_3\text{-}V_3 \wedge A_3\text{-}V_3 \geq A_2\text{-}V_2\ i$
Statements	$<T_1\ A_1\ V_1\ Q_1> \geq <T_2\ A_2\ V_2\ Q_2>$ iff $A_1\text{-}V_1 \geq A_2\text{-}V_2 \wedge \underline{inv}\ A1\_T_1 \geq inv\ A_2\text{-}T_2$

Equivalence In a model M=<*Ps As rps ras Es Ss inv card sb*> equivalence is defined by:

$X_1 \leq \geq X_2$ iff $X_1 \geq X_2 \wedge X_2 \geq X_1$ where X may be a criterion, criteria set, statement or entity.

Sanctioning In a model M=<*Ps As rps ras Es Ss inv card sb*> sanctioning is defined by:

Statements	$sanctioned(<T\ A\ V\ Q>, M) = true$ iff $\forall q \in Qualifiers \; . \; q \geq Q \rightarrow$ $\exists <T'\ A'\ V'\ q> \in Ss. \; <T'\ A'\ V'\ q> \geq <T\ A\ V\ Q>$
Entities	$sanctioned(E, q, M) = true$ i where *M* is a model iff $\forall A\text{-}V \in cs(E).sanctioned(<E\ A\ V\ q>, M) = true$

Coherence In a model $M=<Ps\ As\ rps\ ras\ Es\ Ss\ inv\ card\ sb>$ coherence is defined by three functions, consistency, soundness and irredundancy, defined by:

Consistency *consistent(E,M) =true* where $E=<D\ N\ I>$ is an entity in M iff all of the following hold:

a) $\neg\exists A_1\text{-}V_1, A_2\text{-}V_2 \in ccs(E)\ .\ A_1 \geq A_2 \wedge cardinality(A1)=one \wedge \neg(A_1\text{-}V_1 \geq A_2\text{-}V_2 \vee A_2\text{-}V_2 \geq A_1\text{-}V_1)$

b) $\neg\exists\ A\text{-}V_1\ A\text{-}V_2.$
$A\text{-}V_1 \in ccs(E) \wedge inverse(A)\text{-}V_2 \in ccs(V_1) \wedge$
$cardinality(inverse(A))=one \wedge$
$\neg(A\ V_1 \geq A_2\text{-}V_2 \vee A\text{-}V_2 \geq A\text{-}V_1)$

c)[5] $\neg\exists\ A_1\text{-}V_1\ A_2\text{-}V_2.$
$(A_1<A_2 \vee A_2<A_1) \wedge$
$A_1\text{-}V_1 \in csc(E) \wedge inverse(A_2)\text{-}V_2 \in ccs(V_1)$

Soundness *essentially_sound(E,M) =true* where $E = <D\ N\ I>$ is an entity in M iff
$\forall A\ V.\ <E\ A\ V\ necessary> \in Ss \leftrightarrow A\text{-}V \in N$

inheritance_sound(E,M)=true where $E = <D\ N\ I>$ is an entity in M iff
$[\forall c'\ E'\ .\ E' \in Es\ \&\ E'>E\ \&\ c' \in ccs(E') \rightarrow \exists c \in ccs(E)\ .\ c' \geq c] \wedge$
$[\forall c.\ c \in I \rightarrow (\exists E' \in Es\ c' \in .ccs(E')\ .\ E'>E \wedge c'=c) \wedge$
$(\neg\exists c'' \in N\ .\ c \geq c'')]$

Irredundancy *irredundant(E,M)=true* where $E= <D\ N\ I>$ is an entity in M iff
$\neg\exists E'\ D'\ I'\ .\ E'=<D'\ N\ I'> \wedge D'> D \wedge inheritance\_sound(E',M) \wedge$
$sanctioned(E'M) \wedge consistent(E') \wedge ccs(E) \leq\geq ccs(E')$

5 Condition c) is an added restriction which avoids having to deal with peculiar cases. So far no practical example of such cases has been suggested.

Model Invarants In a model M=<*Ps As rps ras Es Ss inv card sb*> the following invariants are defined:

inverse_complete — *inverse_complete(M)=true* iff
$\forall S \in Ss$. S=<T A V Q> $\wedge$
$Q \in$ {*conceivable, grammatical sensible*)$\rightarrow$
inverse(S) $\in Ss$

essentially_sound — *essentially_soundM) = true* iff
$\forall E \in Es$. *essentially_sound(E,M)=true.*

inheritance_sound — *inheritance_sound(M)=true* iff
$\forall E \in Es$. *inheritance_sound(E,M)=true.*

coherence — *coherent(M)=true* iff all of the following hold :

a) $\forall E \in Es$. *sanctioned(E,M)=true* $\wedge$ *consistent(E,M)=true*

b) $\forall S \in Ss$. *sanctioned(S,M)=true.*

c)[6] $\neg \exists E_1,E_2 \in Es$. E_1=<D_1 N_1 I_1> $\wedge$ E_2=<D_2 N_2 I_2>
$\wedge$ $E_1 \leq\geq E_2 \wedge \neg D_1 \leq\geq D_2$.

irredundancy — *irredundant(M) =true* iff
$\forall E \in Es$. *irredundant(E,M)=true.*

uniqueness — *uniqueness(M)=true* iff
$\neg \exists E\ E' \in Es$. $E' \leq\geq E \wedge E \neq E'$.

properly_structured — *properly_structured(M)=true* iff
inverse_complete(M)=true $\wedge$*inheritance_sound(M) =true* $\wedge$
essentially_sound(M) =true $\wedge$ *coherent(M) =true* $\wedge$
uniqueness(M)=true $\wedge$ *irredundant(M)=true*

Operations

new_prim(p', p, M)$\rightarrow$ *M'* — *M* = <*Ps As rps ras Es Ss inv card sb*> ,
M' = <*Ps' As rp' ra Es' Ss inv card sb*> where

Ps' = *Ps* $\cup$ {*p'*}; *rp'* = *rp* $\cup$ {<*p p'*>}; *Es'*=*Es*$\cup${*E*} where *E*=<*is-P* {} *I*> and *I* is such that *inheritance_sound(E,M)*

new_attr(a', inva', c, c_i,M)$\rightarrow$ *M'* — *M* = <*Ps As rps ras Es Ss inv card sb*>; *c*, $c_i \in$ *Cardinalities;*
M' = <*Ps As' rp ra' Es Ss inv' card' sb*> where

As' = *As* $\cup$ {*a'*}; *ra'* = *ra* $\cup$ {<*a a'*>}; *inv'*=*inv* $\cup$ {<*a' inva'*>};
card'=*card* $\cup$ {<*a c*> <*inva* c_i>}

specialised_by(a_1,a_2,M)$\rightarrow$ *M'* | *error*[7] — *M* = <*Ps As rps ras Es Ss inv card sb*>; *c*, $c_i \in$ *Cardinalities;*

IF $\exists S \in Ss$. S=<T a_1 V q> $\vee$ S=<T *inverse*(a_1) V q>
THEN error
ELSE
M' = <*Ps As rp ra Es Ss inv card sb'*> where
sb'=*sb* $\cup$ {<a_1 a_2>}

---

6 NB. This is a significant restriction. The natural semantics would allow more than one definition, and it is expected that this will be supported in future versions.

7 The restriction to declaring specialisation before any use of an attribute is enforced only by convention in the current implementation.

new_statement(S,M)→
M' | *error*

M = <*Ps As rps ras Es Ss inv card sb*>

M' = <*Ps As rp Es' Ss' inv' card' sb*> where
Ss'=Ss∪*{S}* IF∃*Es'* . *properly_structured(M')=true*
else *error.*

which(E, Cs, M)→
<*E', M'*> | *error*

LET *E* = <*D N I*> , *M*=<*Ps As rps ras Es Ss inv card sb*>

IF ∃ *E''* ∈ *M* . *E''*= <*D'' N'' I''*> ∧ *D''*≥ *D*∪*Cs* ≥ *ccs(E'')* THEN
E'=E'', M'=M;

ELSE IF ∃ *E'' D'' I''* . *E''*=<*D'' {} I''*> ∧ *D''*≥ *D* ∧ *D*≥*ccs(E'')*
inheritance_sound(E'',M) =true∧
irredundant(E'', M) =true ∧
consistent(E'')=true ∧
sanctioned(E'', M) = true
THEN E'=E'', M'=<Ps As rps ras Es ∪*{E''} Ss inv card sb*>

ELSE error