Description Logics And Multimedia - Applying Lessons Learnt From The GALEN Project

Sean Bechhofer and **Carole Goble** [1]

Abstract. We describe some preliminary exploratory work in the use of the GRAIL [2] description logic and GALEN terminology server architecture [24] to support classification of art images, based on our experiences of using GRAIL to support medical applications using clinical terminology [24].
We hope to apply lessons learnt in the medical domain and test whether the decisions made in the design and implementation of GRAIL and the terminology server architecture help or hinder efforts in a different domain.

1 Terminology and Art

The use of terminologies to describe images is not new and predates computers. The principle of descriptive analysis of fine art was expounded by the Prague School in the 1920s [31] in the creation and application of a structured semiotic terminology. The terminology could be used to describe and automatically classify works of art by their content, and moreover identify patterns of change with reference to the social and environmental contexts of the artists and identifying common influences. Such work using picture description languages is highly active in museums [3]. However, manipulating descriptions of works of art that is scalable and flexible requires the use of a terminology such as that provided by the Prague School coupled with a knowledge based computerised terminology able to support the automatic classification of these works.

2 Description Logics

As has been previously proposed [19, 4], multi-media is an area which is ideal for the application of description logics in particular for dealing with the semantic content of objects such as images [30]. Activities such as cataloguing rely on classifications and hierarchies, while the notions of uniform representation and re-use – provided by the use of a terminology – are also desirable.
Description logics, with their subsumption hierarchies, automatic classification and compositional nature provide an ideal representation for metadata – semantic or syntactic information about the content of documents which is required to support content-based image retrieval without reverting to the constant re-interpretation of images [10].
The use of description logics will allow us to support (among other things):

- Incremental elaboration and partial information - concepts can be incrementally specialised, with the automatic classifier of a DL taking care of relationships between concepts. As more information is derived about an image's content, it can migrate down the concept hierarchy;
- Imprecise Querying - the subsumption hierarchy can be used to support imprecise or general queries - the knowledge that the British Prime Minister is a politician holding right wing views can be used in the retrieval of "any pictures containing a politician with right wing views";
- Conceptual Similarity - We may wish to retrieve all video clips of the Prime Minister. If no such clips are available, clips of the Home Secretary may be suitable, but footage of an Under Secretary of State will not do. The concept hierarchy gives us a space through which we can navigate, exploring such relations.

A process which has been identified as important [14] is **annotation** - attaching some structured description or information to an image or document. This raises the issue of data entry - how does the user select or describe the piece of information which is to be attached?

3 A Terminology Server

The GALEN Project has developed the notion of a "terminology server" (or TeS) [24] – a system providing a range of terminological services to client applications. The server provides a central repository for the terminology and associated knowledge and can assist in the process of mediation, giving a unified model. Communication with the TeS is via a well defined Application Programmers' Interface (API), perhaps over a network.

3.1 A Server Architecture

The terminology server uses the model or representation to provide various services to applications. These include:

Conceptual services. Operations that manipulate the model, and answer questions such as "how can I combine these two concepts?", "what sort of thing is this?", and so on.
Language services. An important part of the GALEN work is to produce natural language expressions for concepts, using a dictionary of terms for elementary concepts along with grammar rules describing how to produce terms for complex concepts.
Coding services. There are many existing coding schemes which represent medical terminology. The GALEN project aims to provide mappings between the GALEN model and existing schemes.

Each group of tasks is the responsibility of a different module within the terminology server. Each module has its own data associated with it - the Concept Module has the representation of the model, the

[1] Medical Informatics Group, University of Manchester, Oxford Road, Manchester M13 9PL, UK. {seanb,carole}@cs.man.ac.uk

ECAI 96 Workshop. *KRIMS 96*
Edited by G.Vouros, W.Klas, T.Sellis, M.Vazirgiannis

Language Model has a lexicon of elementary terms and a collection of grammar rules and the Coding Module has representations of other coding schemes and mappings. The modules communicate with one another and the API through well defined interfaces.

The server provides a "wrapping" round the description logic (see below), allowing the representation to be used uniformly by an application, or suite of complementary applications (annotation tools, query tools etc.).

Although the architecture was primarily developed for the support of medical applications, we believe that the approach is applicable to any domain where terminological operations are required. The client-server architecture provides a clean split between the core conceptual and terminological operations associated with the domain model, and the more application specific functionality of a client, for instance an annotation tool.

An important point is that the notion of a terminology server is not tied to the GRAIL language. Although GRAIL has features such as sanctioning (Section 6) which can help to support tools (Section 9), another representation could be used within the server.

4 Motivation

It would be ideal if media objects such as images could be selected and classified (or clustered) such that "conceptually similar" images are grouped together by content. This requires that image content be described by some coherent semantic domain model rather than relying on the use of keywords as in most commercial image database systems [32].

Zdonik [37] proposes that image databases fall into the category of incremental "bottom up" databases, where a description of the images cannot be predefined to fit with a prescriptive database schema as in conventional databases, but must incrementally evolve to link image instances with a schema, or even evolve the schema from describing the image instances. When image data is captured it has little or no form though a good deal of substructure. Applications determine the appropriate substructure, and additional substructure generates more schema. The schema could exist before connections between instances are made, or not. For example, the concept of a UK Prime Minister could exist before images with instances of a prime minister are connected to these concepts. Instances can begin with no annotations linking them to the schema and be incrementally elaborated as more of the content is revealed or is required to be retrieved. So we might have an image of a man that we describe as such, and later elaborate on that description to include an aircraft, and still later further elaborate upon it to name the man as John Major. Hence our instances will always have incomplete or varying completeness of description, and as their descriptions are extended so they are reclassified, (e.g. as an image portraying a vehicle and a politician). Images collect annotations [14], so this kind of incremental support is essential.

Flexibility is required as the same image may be reused from many different application perspectives, and classified and reclassified by many different, unpredictable, and possibly contradictory interpretations of the same contents.

Operations which we should aim to support include:

- Semantic annotations. Annotation with semantic data covering the meaning of images or documents, linking them to a collection of domain concepts;
- Dynamic classification and reuse. Flexible annotations allow the reuse of information from different perspectives - an image of John Major in front of an aircraft may be thought of as an image of a man, a politician, a person by a machine and so on;
- Unpredictability of use;
- Incremental elaboration, reclassification and incompleteness. The use of incremental elaboration brings with it a cost of maintaining the consistency of descriptions.
- Imprecise or similarity retrieval based on annotations. Imprecise or incomplete descriptions of image content will naturally mean imprecise and inexact matching of queries. Query by example, "retrieve all images whose content is similar to this one", is common amongst image-based content retrieval systems [9].

5 Static Hierarchies and Coding Schemes

At least two approaches have been used in order to provide annotations with some structure.

5.1 Keywords

Keyword systems [32] are an attempt to introduce more coherent annotations and can go some way to introducing a common or shared vocabulary. However, they are uncontrolled and tend to provide annotations which are inflexible, difficult to extend and which lack structure.

5.2 Coding Schemes

An improvement on this is to use a "coding scheme" of terms, arranging concepts in a (static) hierarchy then using terms from this hierarchy for annotation. The hierarchy can provide more structure to the annotations. An example of such a scheme is the ICONCLASS system [33, 13], developed as a means for describing the content of works of art. ICONCLASS consists of a large static hierarchy containing entries which have a unique code identifying the concept and its place in the hierarchy, along with a *textual correlate* which is a piece of text describing the concept. As well as the hierarchy, concepts are indexed via keywords, allowing multiple entry points into the hierarchy. ICONCLASS contains around 24,000 definitions including objects, persons, events, situations and abstract ideas.

This mirrors the medical world, where the approach towards terminology was for many years to use static hierarchies such as ICD [21] and SNOMED [7]. However, these hierarchies suffer from deficiencies as detailed in [26] – the schemes are *too big*, in that navigation is difficult, and data entry often results in the use of large "picking lists" ; they are *too small* in that the levels of detail captured is often insufficient.

In order to build usable medical data entry systems, the conclusion of the PEN & PAD [23] and GALEN [25, 36] projects was that a model with a richer structure was needed - a *compositional* model that allowed the terminology to be built up using atomic or primitive concepts, and relations between them.

The hierarchy of ICONCLASS suffers from all the problems encountered in the medical coding schemes; in particular, the semantics of the hierarchy is unclear. For example, beneath the code **41A32** door there are a wide variety of children including:

41A322 closing the door. An action;
41A323 monumental door. A kind of door;
41A3241 door-knocker. A thing found on a door;
41A325 threshold. A rôle a door can play;
41A327 door-keeper, houseguard (inanimate). An object found next to a door.

The hierarchy here is simply *organisational* in that things are placed under one another when there is *some* connection, not necessarily subsumption (the *is-a* relationships which forms the basis for concept hierarchies in description logics). This hampers navigation through the conceptual model as it is not clear where one would find the concept closing the window - is it somewhere under action or under the concept it applies to e.g. window? This organisational interpretation also complicates the task of providing a semantics for the hierarchy.
In addition, static schemes can suffer from "combinatorial explosion". If we consider a model where a door can be made from one of four substances and can have two possible actions performed on it, this can lead to up to fifteen ($(4+1)\times(2+1)$ allowing for the application of no substance and/or action modifiers) codes being required to cater for all combinations. All fifteen codes would then have to be explicitly enumerated.

6 GRAIL

GRAIL (GALEN Representation and Integration Language) is a description logic of the KL-ONE family [34]. It has its origins in the GALEN [25, 36] project, which aims to build a terminology server for the medical domain, and the earlier PEN & PAD [23] project, concerned with providing data entry systems for General Practitioners. GRAIL has been specifically devised for medical terminologies, which has influenced its range of term constructors. Certain constructors found in other DLs, in particular disjunction and negation are not present, and only a limited form of cardinality restriction is provided. GRAIL also differs from its relatives in that it has:

- A **sanctioning** constraint mechanism such that only semantically valid concepts can be combined into descriptions. The sanctioning mechanism allows us to generate concepts implied by the model while guaranteeing their correctness, and can play a major part in the control and construction of data entry mechanisms;
- Essential assertions known as **necessary statements**. These are similar to A-box assertions of other languages, but have terminological significance and play a part in classification;
- A **canonisation** mechanism for ensuring that equivalent and redundant definitions are identified and reduced to a normal form;
- A co-ordination of partitive hierarchies, transitive relations and subsumption, known as **refinement**.

6.1 Sanctioning

GRAIL uses a mechanism known as sanctioning in order to constrain the concept expressions that can be built up. In GRAIL, the philosophy is that no terms can be constructed unless there is an explicit assertion allowing the composition – this differs from the role restriction constructs found in other DLs, where we could use conjunction with a universal quantification to ensure that all the values related via a particular attribute were of some class. This would be a stronger assertion though, precluding the construction of other specialisations. As an example, consider the process of "opening". We wish to restrict the combination of this process with things that can sensibly be opened, thus allowing the construction of "opening a door", but disallowing, for example "opening a carrot". A universal quantification could be used asserting that all processes applied to door must be of kind opening, but we will then be unable to assert that, for instance, the process of painting can also act on a door.

Sanctioning has two levels[2] known as "grammatical" and "sensible", allowing for flexible expression of both general relationships, e.g. Processes act on Structures, and more detailed control, e.g. Opening acts on Containers. Complex expressions cannot be formed until the lower level (sensible) sanctions have been applied, and sensible sanctions cannot be asserted unless a grammatical sanction is present. Sanctioning uses the hierarchy implicitly, with concepts inheriting all the sanctions of their parents. The sanctions above would be expressed as

Process grammatically actsOn Structure.
Opening sensibly actsOnContainer.

The sanctioning mechanism also allows us to use the representation in a generative manner, answering the question "what sort of specializations of this concept could there be?" In an unconstrained system, such a generation of concept descriptions could result in all combinations being developed, including "nonsense". In addition, as the combination and specialisation of concepts is controlled in this structured manner, we can use the structure to guide the construction of interfaces to the representation, leading to tools as described in Section 9.

6.2 Necessary Statements

Necessary statements allow a modeller to make assertions about the properties of a concept. Unlike A-box assertions in other languages, necessary statements have *terminological* significance and change the classification of concepts. If we make the assertion:

(Person which isDriverOf Car) necessarily hasAge old[3]

this alters the classification of the concept (Person which isDriverOf Car) to also be a child of, for instance, (Person which hasAge old), rather then providing a rule which triggers every time a (Person which isDriverOf Car) is recognised (as would happen in, say, LOOM [5]). Note that a necessary assertion does not form part of the definition of concept - we need not know that a car driver is old to recognise the concept.
Necessary statements are indefeasible. Thus in the example above, *all* drivers of cars are old - there are no exceptions.
Necessary statements allow a modeller to introduce concepts and then elaborate their descriptions, rather than asserting all the information at the moment of introduction.
The addition of essential assertions does complicate the process of classification. Early implementations relied on a subsumption test which has been shown to be incomplete. However, recent work has produced complete subsumption tests for the language with essential assertions, and work is continuing in this area [12].

6.3 Canonisation

Canonisation is a mechanism which reduces expressions to a normal form, removing redundancy and spotting incoherent definitions (those with conflicting criteria). Canonisation becomes interesting in the face of necessary assertions. After the assertion introduced above, an expression

(Person which [hasAge old, isDriverOf RedCar]),

[2] At present the formalism uses two levels of sanctioning, but further levels could be added if necessary.
[3] We present examples here using a GRAIL-like syntax.

where RedCar is a kind of Car, would be reduced to

(Person which isDriverOf RedCar),

as the age criterion is inherited from a parent concept.

6.4 Refinement

Refinement has proved to be a useful construct in the medical modelling. The medical model is well populated with partitive and structural relationships which interact with locations. In general, an object located in something which is a part of a larger object is also located in the larger object. Refinement allows us to capture these interactions - in this case we would assert that the relation hasLocation is refined along the hasPart relation.
There are similar relationships in describing image content. A picture containing a man holding a cat is also a kind of picture containing a cat, even though a man holding a cat is not a kind of cat. It is desirable to be able to infer this relationship without having to explicitly note the fact.
GRAIL is still a developing formalism - formal semantics of operations such as refinement have yet to be defined, although an operational semantics in the form of an implementation does exist and has allowed experimentation.

6.5 Limitations

6.5.1 Cardinality

GRAIL only provides a very limited form of cardinality construct - relationships are either *one* or *many*. This restriction was introduced for pragmatic reasons, as more detailed cardinalities (minimum or maximum number restrictions on rôles) introduce further complexity into the subsumption and classification processes. In the medical modelling this was not considered to be too restrictive, but in modelling other domains, this is likely to be a problem. As an example, a concept such as "a person standing beside at least two cars" cannot be expressed easily in GRAIL. However, even in situations such as this, the concept hierarchy still provides a "coarse grained" index which can be used to prune the search space during queries [28].

6.5.2 Value Types

GRAIL has very limited support for value types - strings, integers, dates and so on, allowing the introduction of values such as 2, "green" or 11/4/66. The addition of features like ranges will be required if we are to support a significant "real world" domain model.

7 Modelling

In order to support any of the functionality described above, a comprehensive data model of the domain is needed. This applies equally well to the use of *any* DL, not just GRAIL, and raises several areas of interest. There are different ways of interacting with a terminological model. Our experiences with the GALEN project led to the identification of the following three levels of interaction: building a model, extending a model and using a model.

7.1 Building a new model

For this first task, a model has to be constructed from scratch. This involves many activities, including the identification of a high-level ontology. Producing an ontology is a crucial step in the process if we are to build a *consistent* and re-usable model that can be considered "application independent" – a notion which is highly desirable but often difficult to achieve in practice. The approach in the GALEN modelling [22] is to divide the world into three categories[4]:

Processes – changes which occur over time. Examples of process are the actions opening or closing;
Structures – abstract or physical things with parts independent of time. An example of a structure is door or handle;
Substances – continuous abstract or physical things independent of time. Examples of substances could be wood, metal and so on[5].

To give a simple interpretation, Structures are "things", while Substances are "stuff" that "things" are made out of, and processes "happen" to "things" and "stuff". In addition to the three categories introduced above, we have the notion of:

Modifiers – a heterogeneous grouping including aspects which refine meaning. For example the notions of location (proximal/distal, upper/lower, left/right etc.), shape (round, spherical, laminar, linear etc.) or rôles[6]. We also have so-called *modalities* which take their meaning from a category but which are fundamentally different from the original category, e.g. a *collection* of *fruit* is not a kind of *fruit* but has a meaning related to it.

Although the GALEN high level ontology has been developed in the domain of medicine, it is our belief that most of the ontology is suitable for modelling other domains. Current work includes some exploratory modelling to determine if this is the case - certainly early observations of the semantic categories included in the ICONCLASS classification suggest that the model will fit easily into the high level divisions described above.
The task of building a new model requires a wide range of tools to support it – in particular browsing tools which permit the examination of the content of the model. We also require tools for adding new knowledge to the model. Currently, information is presented in the form of GRAIL "source code", and the principle tool for adding knowledge is the GRAIL "compiler". This is a rather low-level interaction with the model, and we are investigating more sophisticated user tools for model building which allow a modeller to work at a further distance from the basic GRAIL operations. Modellers building new models are required to have a good understanding of the underlying formalism.
The strategy in GALEN was to use an "oracle" approach to modelling, where a collection of experts (in this case clinicians) devise and construct a common reference model. Much use was made of existing corpora such as ICD and SNOMED [21, 7], which provide sources of terms and hierarchies to populate a model. Indeed, if such a classification is to be useful, it must provide similar coverage and overlap to existing systems in use.

[4] The claim is not that this is the "right" way to split the world, simply that it is a way that works – one could choose to reorganise the hierarchy differently, producing a model which was just as effective.
[5] One could argue that wood is in fact some kind of structure - the question is then "how far" we should take the decomposition of the model. With a compositional model it is not always easy to determine when to stop. The intended use of the model is a guide, but this can lead us away from application independent representations.
[6] Not to be confused with the röles or attributes of the GRAIL language.

There are classifications available for images or works of art, such as ICONCLASS, AAT, the CDWA and the terminologies developed by the Prague School [31], but these classifications have incomplete coverage and are strongly application dependent. They should, however, provide suitable sources of terms for an initial modelling effort.

7.2 Extending a Model

A model of a domain is often an evolving object. As new knowledge is discovered, or as new requirements are produced, the model must grow. This activity is different from the task of constructing models as described above. The high level structure is present, and we are involved in the activity of extending or providing further levels of detail. For instance, we may have introduced the notion of gardening implement with some examples, say spade and fork. The introduction of a new implement, e.g. hoe, requires no great change to the structure, and is in a sense a "local" change to the model. Similarly, we may wish to introduce new sanctions, allowing further refinement of concepts, which again has very little effect on the overall structure of the knowledge. Tools can be provided which help to move the modeller away from the underlying description logic, reducing the need for modellers to be completely *au fait* with the formalism.

Note that although related, this is not the same as the problem of elaborating instances. When elaborating instances, for instance asserting that a particular picture contains an open door, we are *using* the conceptual structure. Extension involves adding new information to the conceptual model.

7.3 Using the Model

The final task involves the *use* of the model. This covers activities such as annotating images, retrieving images, performing queries using the structure and so on. During this process, the underlying knowledge in the model is not changed. This task requires particular tools, and we have some experience of providing data entry tools which aid in the construction of concept expressions and allow the user to move away from the underlying GRAIL "source" [1, 17]. This is of course desirable if users with *no* experience or interest in the syntax and details of the representation are to use the model.

8 Consistency

The use of partial information and incremental elaboration raise problems of consistency. How much re-organisation must be done to the classifications of instances when new information is added? In a highly interconnected model, this work becomes harder as the model grows. Although completeness of the system is desirable, how much is pragmatically required to implement a useful system?

We can consider maintaining differing levels of consistency according to the annotations we wish to make and the amount of information that may be added to those annotations. These can be separated into three basic levels.

No Refinement Media objects are simply given a description which will not change in the future.

Topic Refinement Only media objects can have their descriptions further refined. After asserting that a picture contains a red car, we may later add that there is also a person standing in the picture. Hence, we may need to reclassify the objects representing the images, as their descriptions have changed (this was a picture containing a car, and is now a picture containing a car and a person).

Arbitrary Refinement Finally, we can allow everything (including domain world objects) to be refined and further described. Thus after asserting that an picture has as its subject an individual John Major, we may add the knowledge that John Major is in fact the Prime Minister of the UK. The addition of this (domain) information causes a reclassification of the individual John Major, and should also cause reclassification of the image, as it is now an image not only of a person, but of a Tory Politician.

In the first two schemes we explicitly forbid further refinement of the *world* objects in the conceptual model – in the third it is allowed, but will incur a cost.

Of course even with the simplest of the schemes described above, the hierarchy is still available for query using subsumption and inheritance. Even though all the information introduced about an image is "static", we can use imprecise querying. In an example above, an image is asserted to be an image of a red car. Subsequent searches for all images of vehicles will successfully return this image.

We examine each of the above schemes more closely.

8.1 No Refinement

Once an image has been described, no further information can be added - either to the domain or to the image itself. When an instance is introduced, it has associated with it the concept of which it is an instance.[7] This association will never change.

If we assert that a particular image is a picture of a red car, we can say nothing more about it at a later date - no refinement of the information is possible. This is the simplest form of annotation, and in this scenario, very little work is required to maintain the consistency of the information.

As discussed in Section 4, incremental elaboration is essential. Thus the highly restrictive scheme is unlikely to provide much utility.

8.2 Topic Refinement

We can make a distinction between *world* or *domain* objects and *application* objects. Application objects are the representations of the objects which we wish to annotate – our images, videos, pictures, etc. World objects are those things in the domain which feature as content of the images.

In this scheme, we are only allowing assertions to be made about application objects. This takes a somewhat application-dependent view of our interaction with the model – we can make assertions about a particular class of things that we know we're interested in doing things with (annotation, retrieval, etc.). Although both the world and application objects sit in the same model, we have separated them (in the sense that different operations are allowed on the two classes of things).

This allows more than the restrictive scheme, but still prevents addition of domain knowledge. In this scenario we also have to be aware of attempts to treat application objects as domain objects – examples such as "this is a video featuring a picture", where the picture has been described.

Annotation here is a one-way process. By asserting that picture 17 contains John Major, we change the classification of picture 17, but do *not* change the classification of the instance John Major. Thus queries such as

> "Which pictures contain Tory politicians?"

[7] We may also wish to note for each concept which instances are instances of it to aid in querying

will return picture 17, but a query such as:

"Which politicians feature in pictures?"

cannot be answered by simply using the concept hierarchy.[8]

8.3 Arbitrary Refinement

When we allow arbitrary addition of information, maintaining consistency is difficult. Any change to any object, be it a world object or an application object can cause reclassification of other objects throughout the model. This leads to a propagation or "rippling" of reclassification which can potentially effect the entire world of individuals – an undesirable situation.

The problem of maintaining consistency and ensuring that all inferences are found also appears in other DLs, for example LOOM [5]. The situation is made worse here by the inclusion of inverse rôles (in GRAIL all relations have inverses). When we assert that John owns a cat Tom, then we are implicitly asserting that Tom is owned by John. The question really is how far can we get with restrictive schemes – can they provide enough flexibility? We return to this point in Section 11.

8.4 A Lazy Index

If we adopt any of the schemes above where refinement or specialisation of annotations is permitted, annotation is the more expensive activity. We are using the description logic as an "index" for the contents of images.[9] The classifier and concept hierarchy give us what is effectively a "precompiled" index of all possible queries. As an example, consider a request for any images containing a picture of a male politician. The classifier will determine the position in the hierarchy of the concept image which contains male politician, and thus can determine all instances which "fit" this description. Note that if this concept has not been previously requested, the concept's position can still be determined by the classifier – thus the hierarchy can answer questions which have not been presented (or even formulated) before. It is because of this property of the hierarchy – already having the answer to "any" question – that annotation is difficult, as we have to ensure that all relationships are up to date when adding new information to the model. This is the source of the "ripple" problem with the Arbitrary Refinement scheme described above.

9 Tools

Along with the increased flexibility and power of a compositional scheme and shared vocabulary or terminology come the problems of access to that vocabulary.

Once a domain model has been constructed, we need tools to help in the task of interacting with the model, particular for the two tasks of:

Annotation The process of attaching information to objects. For instance, we may wish to annotate a picture of John Major with the fact that this is a image of a British Tory Politician;

Querying Having annotated objects, we want to retrieve them based on the annotations we've made – for example "get me a picture of a right wing man".

[8] We could of course answer this question by examining all the instances of politicians, but that's not the point here.

[9] Although we're using images here as the example, the discussion extends equally well to video, documents, etc.

In order to perform either of these tasks successfully, we need to be able to:

1. Determine what there is in the domain model. Ideally we would expect the model to cover all the domain, but in reality there may well be areas where coverage is not complete and a "close" concept must be chosen.
2. Construct suitable expressions. This of course only applies to a compositional scheme.

These can be paraphrased as the two questions "what things are there in the model" and "how can I put them together?"

9.1 Browsing

Browsers provide some support for examining what is contained in the model. Figure 1 shows a GRAIL browser focussed on the concept of door (a concept represented in the ICONCLASS hierarchy). The

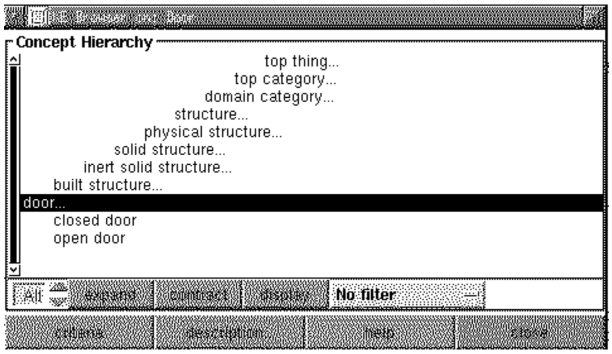

Figure 1. Model Browser

browser provides a simple window onto the concept hierarchy - we can see the parents of door along with a couple of specialisations. More graphical displays were initially used, but these proved too unwieldy once the model grew large, so the hierarchy is shown in a textual manner. The names displayed in the browser are not GRAIL expressions, but are natural language expressions which have been generated from the concept definitions. In the case of door and its parents, these are simply strings which are attached to the concepts. In the case of open door, the string is generated from the information attached to the concept representing door, the concept representing the state of open-ness, and a rule describing how to fit the two together. The browser also illustrates the shape of the higher levels of the model. Much of the high-level ontology is asserted - in much the same way as happens in a traditional coding scheme. However, once we reach a certain level of detail (in this case door), the compositional nature of the model comes into play, allowing the construction of variants of door without the need to pre-specify them or their place in the hierarchy. It is this behaviour that can aid us in supporting incremental elaboration and similar operations.

Figure 2 shows an additional pane of the browser, indicating the constraints which have been sanctioned for the concept door. The sanction relating to action has been highlighted, showing that door can be the subject of an OpeningClosingAction. The sanction has been given a qualifying level of sensible – the lower of the two sanctioning levels introduced earlier, allowing the combination of door

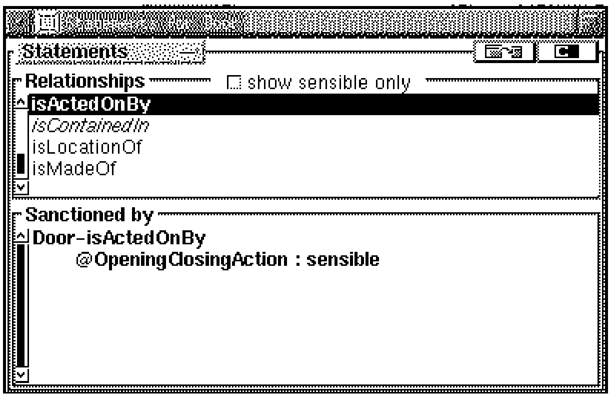

Figure 2. Model Sanctions

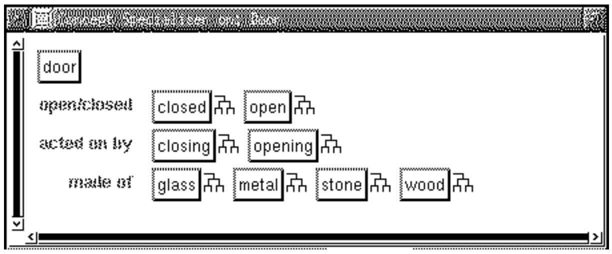

Figure 3. Model Interaction Form

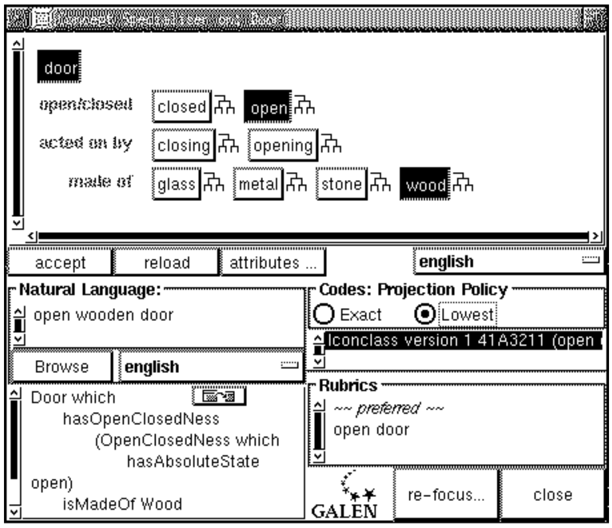

Figure 4. Selection of a concept

with the action. The concept OpeningClosingAction is an abstract concept, encompassing the notion of either opening or closing. By using such abstract concepts, the sanctioning mechanism can be used sparsely.
The browsers have additional functionality which allows navigation round the model.

9.2 Construction

Although useful for those involved in the modelling process, the browsers provide a very "implementation based" view of the model, and require some knowledge of the underlying structure. In addition, they don't provide mechanisms for constructing new concept definitions.[10]
As the construction of new concepts is constrained by the sanctioning, we can use the presence of sanctions to drive this construction process. Consider the example of a door as introduced earlier. In the example model, we consider that a door can be made from some particular substances, and can be acted upon by processes such as opening and closing. These constraints are expressed via sanctions as demonstrated above. This information can then be used to produce "forms" allowing construction of concept definitions. Figure 3 shows such a form, which can be included in an annotation tool.
The look and feel of the form is entirely the responsibility of the form-building application. However, the content (i.e. what is displayed) is dictated by the model, with the information being provided by the server.
As discussed in Section 5.2, presenting this information using a static scheme would require a screenful of codes, each differing only slightly from each other.
Figure 4 shows the situation once the required options have been selected. The appropriate concept definition can be constructed and classified by the server, producing a concept reference and a language term. The bottom left pane contains the GRAIL expression representing the concept, while the pane above contains a language string generated for that concept.
The panels on the bottom right contain information about other coding schemes (in this case ICONCLASS). The server has detected a correspondence between the concept selected and the ICONCLASS classification and this is shown here. The code is shown in the upper pane while the rubric (or "textual correlate" to use the ICONCLASS terminology) is shown in the lower pane.
The correspondence is not exact though - the ICONCLASS code describes "open door", while the concept selected is a specialisation of it. By using the "lowest" matching policy we look for any codes attached to the nearest ancestor of the concept in the hierarchy - in this case finding one associated with "open door". Although the compositional model is more detailed than the coding scheme, we can still relate back to the codes, helping us to find an appropriate code in situations where there is incomplete coverage. By using mappings in this way we can use the predictive data entry techniques as a structured interface to the static classification.

9.3 Mediation

As the mappings are bidirectional between external codes and their corresponding concepts, the terminological model can be used for mediation between differing coding schemes. The matching policies described above provide a mechanism which allows conversion to "approximate" codes when no direct equivalent is present in another

[10] The phrase constructing new definitions refers to the activity of assembling existing pieces of model rather than adding new knowledge to the model. Thus we are in our third level of interaction as described above - *using* the model

scheme. In addition, by using a central mediating model, we need only supply two conversions for each new scheme. This does require that the central representation has at least as much structure and content as the coding schemes.

10 Related Work

A common approach is to code the semantic information in some type structure, for example by using object-oriented methods [15]. Although object oriented systems provide many suitable features for multimedia database systems, Oomoto and Tanaka [20] in particular make the criticism that OODB type systems are generally static and do not support schema evolution well. They propose a descriptive schema that is evolutionary but within the framework of a conventional OO approach that doesn't support automatic classification. Lahlou [16] shares many of our aims and uses a Semantic Data Model to describe images; however his model doesn't appear to support automatic class classification. MORE [35] supports multiple views on the same instances by using domain knowledge to enhance the media instances' OO type system with pseudo objects that are media-specific and derivable from the instances, including content analysis functions. The inferencing is not terminologically based and the type hierarchy, including the pseudo attributes, needs to be explicitly asserted. Our approach would extend MORE with a more sophisticated concept model.

Frame-based systems [29], of which DLs are a principled form, are more flexible. Many authors [11, 30] have used some form of knowledge base, usually based on semantic nets or frames, to describe images, drive image interpretation systems or to automatically label features with a semantic description, but without directly exploiting the imprecise querying and automatic classification possible through the use of a DL or using the knowledge descriptions directly as an instance annotation mechanism. PaVE [27] provides natural language processing but uses a static terminology for art. DLs have been used in the field of Information Retrieval to describe and classify documents [19].

11 Conclusions and Discussion

From these early experiences, DLs appear to be promising with regard to describing, annotating and retrieving semantically media documents. We plan to extend GRAIL to cope with some of the deficiencies described in section 4 (e.g. first class support for numbers) and experiment with other DLs, in particular we shall investigate the support that DLs can offer structural annotations such as spatial and temporal relationships.

Annotation of images is a major effort. Producing annotations entirely "by hand" is not a viable strategy – automated help is required. This can involve either automatic analysis of the syntax of images through the use of non-textual descriptions such as histograms and signatures, or, as in the case of systems such as PICTION [30], the additional use of natural language processing to analyse captions attached to images. Many libraries of images have associate cataloguing information which can help in the production of annotations for those images. In order to automate the process we need to take textual or verbal descriptions, deconstruct them and create the corresponding GRAIL concept definitions. The MultiTale tools produced by the ANTHEM [6] project have been used to build mappings from medical procedure codes to GRAIL, and the presence of the terminological model can be of use during this natural language analysis.

Well-structured data entry is essential when producing annotations by hand – the techniques described in the paper can provide tools allowing access to the terminological model which can facilitate annotation.

The medical modelling effort has suggested a high level ontology as described earlier in the paper. A question is whether this shape fits with models of other domains. As illustrated in Section 9, our initial experiences are that the ontology will be useful, but further experimentation is required. The "oracle" model is suited to medicine where there is a general consensus on terminology[11] but there are questions about whether such a model can be constructed for other domains. The existence of terminologies for the content of works of art suggests that this is a domain for which there is some consensus. Existing coding schemes and hierarchies can be used as sources of terms for the construction of models – for large hierarchies automated support in this process is desirable.

The flexibility of incremental elaboration brings with it the cost of maintaining consistency. One possible approach here is to allow a certain degree of inconsistency to be present in the model. When assertions are made, we could choose not to reclassify, or provide some threshold or distance beyond which no reclassifications are made. We can then return to the hierarchy and perform some batch reclassification, bringing the relationships up to date. It is not clear where this threshold would need to be to produce a usable system. In order to determine this we need further investigation of the ways in which elaborations and annotations are made – "profiling" information which can aid in the optimisation of these operations in an analogous fashion to techniques used in traditional database implementations.

A hope is that the use of a central vocabulary or terminology along with the server architecture can aid in mediation, using the rich structure of a description logic representation to translate between differing information sources. This has been explored with some success for the medical domain [8] and is the subject of the TAMBIS [18] project, which is investigating the usage of the terminology server technology to mediate between biological information sources.

Migration of models is an additional concern. In the database world, migrating data after a schema change is often a painstaking process, frequently necessitating hand-crafted conversions. Can the use of a compositional scheme such as that proposed here aid in this process?

REFERENCES

[1] L.L Alpay, W.A. Nowlan, W.D Solomon, C. Lovis, R.H. Baud, T. Rush, and J-R. Scherrer, 'Model-Based Application: The GALEN Structured Clinical User Interface', in *Proceedings of AIME*, (1995).

[2] S. K. Bechhofer and W. D. Solomon, 'A Tutorial Introduction to the GRAIL Kernel'. GALEN Documentation, Vol. C. University of Manchester.

[3] H. Besser, 'Visual access to visual images: The UC Berkeley image database project', *Library Trends*, (1990).

[4] A. Borgida, 'Description Logics in Data Management', *IEEE Trans Knowledge and Data Engineering*, (1995).

[5] D. Brill, *Loom reference manual version 2.0*, USC, Information Sciences Institute, Dec 1993. Available from: `http://www.isi.edu/isd/LOOM/LOOM-HOME.html`.

[6] W. Ceusters, G. Deville, E. Herbigniaux, P. Mousel, O Streiter, and G. Thienpont, 'The anthem prototype', Technical Report 31, IAI, Saarbrücken, Germany, (1994).

[11] We are not claiming here that clinicians agree on clinical matters, but they are being provided with a forum within which they can disagree.

[7] R.A. Cote, D.J. Rothwell, J.L. Palotay, R.S. Beckett, and L. Brochu, *The Systemised Nomenclature of Medicine: SNOMED International*, Colleg of American Pathologists, Northfield IL, 1993.

[8] P.J. Crowther, 'A Guide to GALEN's Coding Services'. GALEN Documentation, Vol. J. University of Manchester.

[9] M. Flickner, H. Sawhney, et al., 'Query by Image and Video Contents: The QBIC System', *IEEE Computer*, (1995).

[10] C.A. Goble, C. Haul, and S.K. Bechhofer, 'Describing and Classifying Multimedia using the Description Logic GRAIL', in *Proceeedings of Conference on Storage and Retrieval of Still Image and Video IV, San Jose*, volume 2670. SPIE, (1996).

[11] C.A. Goble, M.H. O'Docherty, P.J. Crowther, M.A. Ireton, J. Oakley, and Xydeas C.S., 'The Manchester Multimedia Information System', in *Advances in Database Technology EDBT'92, Third International Conference on Extending Database Technology, Vienna*, (1992).

[12] Ian Horrocks, 'Subsumption Algorithms for the GRAIL description logic'. Unpublished Note, 1995.

[13] IRDG. Iconclass WWW home page. `http://www.let.ruu.nl`.

[14] R. Jain, A.P. Pentland, and D. Petkovic. Workshop Report: NSF-ARPA Workshop on Visual Information Systems. available from UC at San Diego, USA, 1995.

[15] W. Klas, E.J. Neuhold, and M. Schrefl, 'Using an object-oriented approach to multimedia data.', *Computer Communications*, **13**(4), 204–216, (1990).

[16] Y. Lahlou, 'Modelling complex objects in content-based image retrieval', in *Proc Storage and retrieval for Image and Video Databases III, San Jose, CA, USA*, (1995).

[17] University Of Manchester. SCUI WWW page. `http://www.cs.man.ac.uk/mig/galen/scui.html`.

[18] University Of Manchester. TAMBIS WWW home page. `http://www.cs.man.ac.uk/mig/tambis/index.html`.

[19] C. Meghini, F. Sebastiani, U. Straccia, and C. Thanos, 'A model of information retrieval based on a terminology logic.', in *Proc ACM SIGIR93, Pittsburgh, USA*, (1993).

[20] E. Oomoto and K. Tanaka, 'OVID: Design and Implementation of a Video-Object Database System.', *IEEE Transactions on Knowledge and Data Engineering*, (1993).

[21] World Health Organisation, *International Classification of Diseases: Ninth Revision*, World Health Organisation, 1977.

[22] A. L. Rector, J. E. Rogers, and P. Pole, 'The GALEN High Level Ontology'. Submitted to MIE 96.

[23] A.L. Rector, B. Horan, M. Fitter, S. Kay, P.D. Newton, W.A. Nowlan, D. Robinson, and A. Wilson, 'User Centred Development of a General Practice Medical Workstation: The PEN & PAD Experience.', in *Proceedings of Computer Human Interaction CHI '92, ACM Conference on Human Factors in Computing Systems - Striking A Balance, Monterey*, (1992).

[24] A.L. Rector, W.D. Solomon, W.A. Nowlan, T.W. Rush, P.E. Zanstra, and W.M. Claassen, 'A Terminology Server for Medical Language and Medical Information Systems.', *Methods of Information in Medicine*, (1995).

[25] A.L. Rector, P. Zanstra, W.D. Solomon, and the GALEN Consortium, *Health in the new Communications Age*, chapter GALEN: Terminology Services for Clinical Information Systems, IOS Press, 1995.

[26] J.E. Rogers and W.D. Solomon, 'Putting the Clinical into Clinical Information Systems', in *Proceedings of Primary Health Care Specialist Group, Cambridge*, (1995).

[27] L. Rostek and W. Mohr, 'An Editor's Workbench for an Art History Reference Work', in *Proc ECHT'94, Edinburgh, UK,*, (1994).

[28] A. Schmeidel, 'Persistent Maintenance of Object Descriptions using BACK.', Technical Report KIT Report 112, Dept of Computer Science, Technische Universität Berlin., (1993).

[29] S.W. Smoliar and H. Zhang, 'Content-Based Video Indexing and Retrieval.', *IEEE Multimedia*, (1994).

[30] R.K. Srihari, 'Automatic Indexing and Content-Based Retrieval of Captioned Images', *IEEE Computer*, (1995).

[31] Titunik, *Semiotics of Art*, MIT Press, 1973.

[32] H. Treat, E. Ort, J. Ho, M. Vo, J-S. Jang, L. Hall, F. Tung, and D. Petkovic, 'Searching images using Ultimedia Manager', in *Proceedings of Conference on Storage and Retrieval for Image and Video Databases III*, volume 2420, pp. 204–215. SPIE, (Feb 1995).

[33] H. van de Waal, *ICONCLASS. An iconographic Classification System*, Koninklijke Nederlandse Akademie van Wetenschappen, 1973–1985.

[34] W.A. Woods and J.G. Schmolze, 'The KL-ONE Family.', *Computers Math. Applic.*, (1992).

[35] A. Yoshitaka, S. Kishida, M. Hirakawa, and T. Ichikawa, 'Knowledge Assisted Content-Based Retrieval for Multimedia Databases', *IEEE Multimedia*, (1994).

[36] P.E. Zanstra, A.L. Rector, W.D. Solomon, T. Rush, W.A. Nowlan, and S.K. Bechhofer, 'A terminology server for integrating clinical information systems: the GALEN approach.', in *Proceedings of Current Perspectives in Healthcare Computing, Harrogate*, (1995).

[37] S.B. Zdonik, 'Incremental Database Systems: Databases from the Ground Up', in *Proc ACM SIGMOD93*, (1993).