Interpreting Line Drawing Images: A Knowledge Level Perspective

Tony P. Pridmore, Ahmed Darwish and Dave Elliman

Image Processing & Interpretation Research Group
School of Computer Science and Information Technology
University of Nottingham, Nottingham, UK
{tpp,axd,dge}@cs.nott.ac.uk

Abstract. Image understanding systems rely heavily on a priori knowledge of their application domain, often exploiting techniques developed in the wider field of knowledge-based systems (KBSs). Despite attempts, typified by the KADS/CommonKADS projects, to develop structured knowledge engineering approaches to KBS development, those working in image understanding continue to employ unstructured 1st generation KBS methods. We analyse some existing image understanding systems, concerned with the interpretation of images of line drawings, from a knowledge engineering perspective. Attention focuses on the relationship between the structure of the systems considered and the KADS/CommonKADS models of expertise, sometimes called generic task models. Mappings are identified between each system and an appropriate task model, identifying common inference structures and use of knowledge. This is the first step in the acquisition of models of the expertise underpinning drawing interpretation. Such models would bring significant benefits to the design, maintenance and understanding of line drawing interpretation systems.

1 Introduction

Early knowledge-based, sometimes called expert, systems typically comprised large collections of simple, supposedly independent, rules. These were applied to input data by a monolithic inference engine. It was often argued that this architecture allowed expert systems to make flexible use of their knowledge, be developed incrementally and/or provide explanations of their reasoning. In practice the unstructured nature of first generation expert systems made them extremely difficult to develop, maintain and understand. In recent years significant attempts have been made to replace the ad hoc tools and techniques underlying classic expert systems with a more structured knowledge engineering approach. A number of knowledge engineering methodologies have appeared, though KADS/CommonKADS [1-4] is arguably the

D. Blostein and Y.-B. Kwon (Eds.): GREC 2002, LNCS 2390, pp. 245-255, 2002.
© Springer-Verlag Berlin Heidelberg 2002

best developed. Central assumptions of the model-based, knowledge-engineering approach are:

- the simple "rules + inference engine" model masks, rather than describes, the true structure of knowledge-based systems
- system and inference structures are naturally closely related to task type.

Key to these methodologies is the notion of the generic task model or model of expertise, which describes the knowledge, inference and control structures that may be used to perform instances of particular classes of task. An early step in the development of a system using KADS/CommonKADS is to identify which of a library of task models is most applicable to the problem at hand. Subsequent design decisions are then made with reference to this template, which may be modified if the task does not fit an existing model with sufficient accuracy. Access to a suitable model of expertise allows the system developer to be clear about the function of each of the available knowledge sources. System failures (and successes) can therefore be more easily attributed to specific system components. User interfaces are also easier to develop when the inference structure of the task is known and it is clear what knowledge the user is required to provide at each point in the system's operation.

Knowledge engineering has had a significant impact on KBS development and subsequent industrial uptake. While KADS/CommonKADS is very much a European methodology, similar issues have been extensively investigated in the US by, e.g. Chandresakaran [5]. Knowledge-based image understanding systems, however, continue to rely upon 1st generation techniques and design methodologies; unstructured rule bases remain the norm. Crevier and Lepage [6], for example, use the standard rules + inference engine architecture as the basis of their review.

In the longer term, the research reported here aims 1) to examine the possibility of applying knowledge engineering to the development of image understanding systems and 2) to produce line drawing interpretation systems based on sound knowledge engineering principles. We begin by analysing existing drawing understanding systems from a knowledge engineering perspective; identifying common inference structures and use of knowledge. In this we follow Clancey's [7] seminal work on medical diagnosis systems. Attention is focused throughout on the relationship between the structure of the systems considered and the existing models of expertise.

2 Models of Expertise: Classification vs. Configuration

Line drawing interpretation comprises two closely related tasks. A successful system must both determine the type of each entity present and recover a sufficiently accurate spatial description, ensuring that each reported entity is correctly located, both in absolute coordinates and in relation to neighbouring entities. These two issues of classification and spatial coherence are, for example, both explicitly addressed by current [8] performance evaluation schemes. Initial examination of the KADS/CommonKADS model library suggests that, given these goals, models describing classification and configuration tasks are the most relevant.

Classification models are perhaps the most commonly used elements of the KADS/CommonKADS library, and have already been used to some extent in image analysis. Ton et al [9] explicitly based their Landsat image segmentation method on a

form of classification known as systematic refinement, though a recent review [10] of aerial image interpretation systems suggests that heuristic classification [3] is more commonly adopted, albeit implicitly, in that domain. Configuration models are intended to describe design tasks in which the goal is to assemble some artifact given a catalogue of available components and a design brief in the form of a set of requirements on and constraints between components. To apply this model to drawing interpretation would be to take an analysis by synthesis approach.

In the remainder of this section we introduce generic task models of classification and configuration. CommonKADS grew from a series of EU projects active throughout the 1980s and 90s. Over this period a number of notations and formalisms were developed and used to describe models of expertise. While the UML-based notation described by Schreiber et al [4] is the most recent, we adopt, for their simplicity and clarity, the conventions employed by Tansley and Hayball [3].

2.1 Heuristic Classification

A generic task model comprises an inference structure, task structure and strategies. Few KADS/CommonKADS task models, however, include detailed strategy layer; strategy knowledge, though high level, is often also strongly domain dependent. The inference structure is a diagram in which rectangles represent domain roles - the knowledge and data available to and manipulated by the system - and ovals denote inferences over those roles. Arrows highlight possible chains of inference. These diagrams are supplemented by descriptions of each domain role and inference type. In heuristic classification (Fig. 1), for example, the role observables contains "any observable phenomenon". Variables represent "the value placed on the observed data, from the system's perspective". Solution abstractions are "an abstract classification of a problem (or other concept)" and solutions are "a specific, identified solution".

Each inference type description contains a brief textual description of its inputs, outputs, knowledge used/required and possible implementation methods. During Abstract "observable data are abstracted into variables". This requires definitional knowledge and may be achieved via definitional abstraction, based on essential and necessary features of a concept, qualitative abstraction, a form of definition involving quantitative data, or generalization in a subtype hierarchy [3]. Match operates "heuristically by non-hierarchical association with a concept in another classification hierarchy". This may be achieved using heuristic rules. Finally, Specialise refines abstract solutions into more specific ones, again exploiting heuristic knowledge [3].

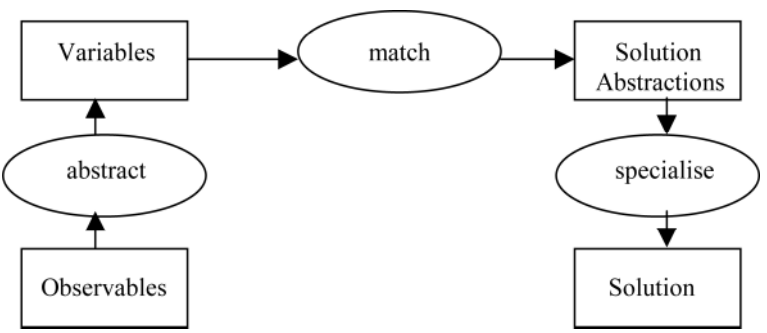

Fig. 1. The inference structure of Heuristic Classification [3]

The task structure component of a generic task model is a pseudo-code description of ways in which the inference structure might be traversed. The task structures for the two most common forms of heuristic classification are [3]:

```
/* Forward Reasoning */
Heuristic Classsification (+observations, - solutions) by
   Obtain data (for observables)
   Abstract (+ observables, -variables)
   Match (+variables – solution abstractions)
   Specialise(+ solution abstractions, - solutions)

/* Backward Reasoning */
Heuristic Classsification (+observations, - solutions) by
   Specialise(-solutions, - solution abstractions)
      Match (-solution abstractions, -variables)
         Abstract(-variables, +observables)
            Obtain data (for observables)
```

Heuristic classification is well suited to line drawing interpretation: it relies on hierarchical representations of both data and solutions and employs heuristic knowledge in the match inference. The model was, however, designed with symbolic, not iconic data in mind. If it is to be employed within line drawing interpretation some way must be found to incorporate the spatial dimension of the problem.

2.2 Incremental Configuration

Configuration is a simple form of design in which components must be selected from a fixed catalogue and placed into an empty or partially filled framework so as to satisfy a set of a priori requirements and constraints. In systems based on incremental configuration, initial requirements are broken down to form descriptions of the resources required by each design element and the constraints on how they can be arranged within the system. Element demands and constraints are then used to identify grouping contexts, usually functional areas or sub-systems of the design, which in turn are used to select a set of elements from the current design to which additional elements need to be connected. In a separate branch of the inference structure (Fig. 2), element demands and constraints are also used to identify components that would satisfy the requirements if included in the design. The required design elements are then compared to those identified as needing extension, any additional required design elements are identified and an Assemble inference employed to merge the new elements into the existing design.

As a model upon which to base drawing interpretation, incremental design is attractive. It neatly separates spatial issues (which entities to extend and in which grouping context) from the classification problem of deciding what type of design element (drawing entity) should be included. One would expect a drawing interpretation system based around this task model to similarly identify and separate these key components. The effective use of constraints is a key issue in scene formation; incremental design provides one template for their use. One criticism that

might be made of the model, however, is that it treats design elements/drawing entities as flat sets of unrelated constructs. The hierarchical data and solutions often found in both classification tasks and drawing interpretation has no place in Fig. 2.

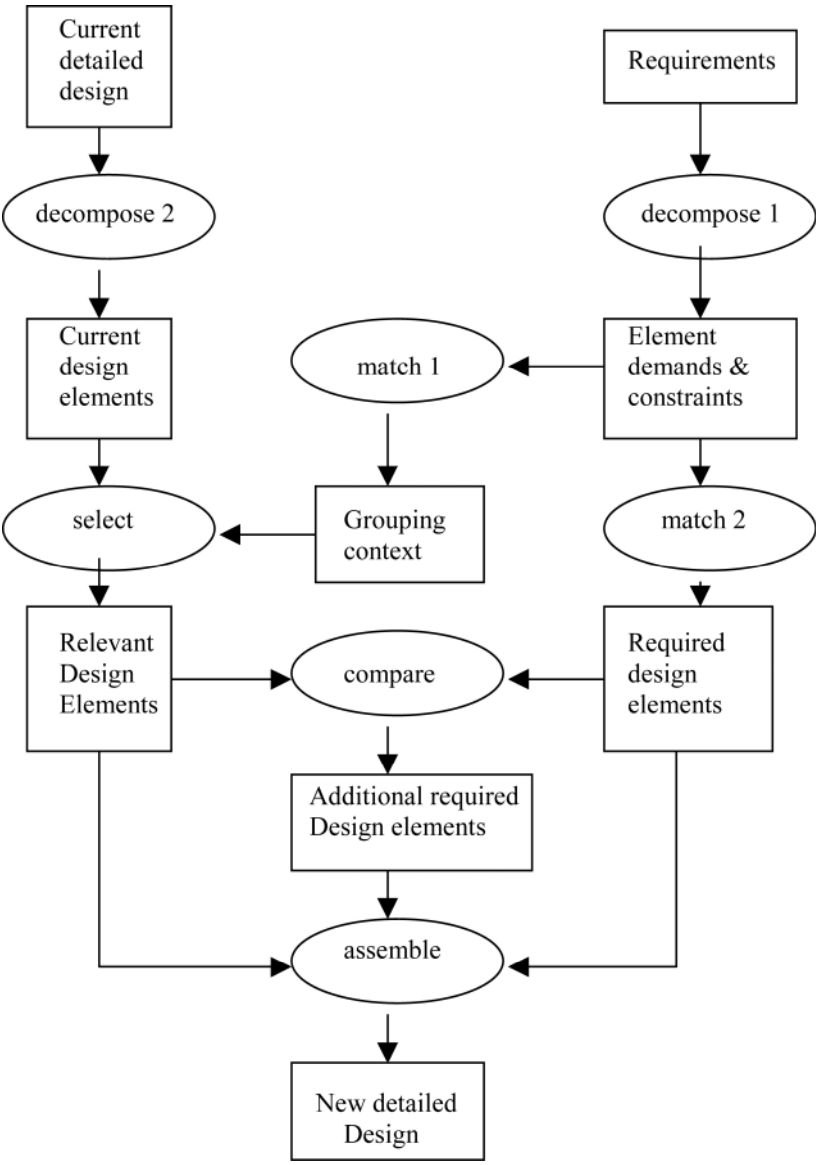

Fig.2. The inference structure of Incremental Configuration [3]

3 Mapping Rule-Based Systems to Expertise Models

One of the most direct attempts to exploit the rule-based expert system paradigm in drawing interpretation is the SAFE system of Goodson and Lewis [11]. SAFE aims to extract extended curvilinear features from images of sea charts. The system is semi-automatic, but relies heavily on a rule-based KBS, only querying the user when the KBS cannot provide a satisfactory response.

The initial image processing operations (Gaussian filtering, thresholding) may be thought of as a KADS/CommonKADS Transform inference. While Transform is defined by Tansley and Hayball as converting "one structure into another", it is commonly used to model general algorithmic operations. The subsequent extraction of line segments best corresponds to the Aggregate inference, which "takes an unstructured or partially structured collection of objects and results in a more or completely structured arrangement of the objects". In this context the objects are pixel locations and the structure imposed aggregates those objects into vector segments.

SAFE's KBS is a backward chaining rule-based system operating over a working set of symbolic statements about lines and their relationships. Each statement is assigned a certainty value that is updated, using a method similar to MYCIN, as interpretation progresses. As is often the case with 1st generation expert systems, the available written description of SAFE considers the statement and rule formats and certainty propagation method, but does not discuss the operation of the system at the knowledge level, i.e. in terms of inference and task structures, etc. The examples given strongly suggest, however, that the IKBS performs heuristic classification, matching the line segments provided by the Aggregate inference to a set of (probably implicitly represented) drawing entities. It is not clear whether any Specialisation occurs, though this seems likely.

Described at the knowledge level, SAFE's KBS performs a variant of heuristic classification in which the Abstract is replaced by an Aggregate and an additional Transform inference is included. The task structure, assuming a Specialise stage, is

SAFE (+image, -features) by
  Transform (+image, -binary image)
  Aggregate(+binary image, -line segments)
  Specialise (+feature, -feature class)
  Match(+line segments, +feature class)

the result of the match being to test for the presence of a particular feature type.

Although less obviously influenced by classic expert systems such as MYCIN, MARIS [12] also takes a rule-based approach. MARIS extracts building lines, contour lines and lines representing railways, roads and water areas from images of maps. The input image is vectorised by a process which labels pixels with line widths, thins, creates and prunes a graph structure and approximates pixel strings with straight lines. Rules are then applied, in fixed order, to identify the constructs sought.

The inference structures of MARIS and SAFE are very similar. There is no initial Transform operation, in the MARIS software at least, as the scanner employed produces a binary image. The Transform is, however, implicit in the complete system. Examination of MARIS' rules shows clear examples of both heuristic Match and

Specialisation being performed. Buildings are identified by thresholding their internal areas, border lengths etc. Continuous lines are extracted and specialised to contours.

Both SAFE and MARIS identify drawing features by collecting together and performing tests on groups of line segments. It is therefore tempting to view the Match inference in both systems as also comprising some measure of Aggregation. The key operation of Match, however, is to make a decision; to label each vector as contributing to one of a predefined set of drawing constructs. This decision is made after consideration of groups of vectors, in the same way as a medical diagnosis system considers the existence and severity of a range of symptoms when deciding whether or not the patient is suffering from a virus. Any vector grouping that occurs during this stage is therefore contained within and secondary to the Match inference. It is often difficult to see the boundary between Aggregate and Match in rule-based systems like SAFE and MARIS, but in inference terms the separation is clear.

Recognition that these two systems are instantiations of the same generic task model allows them to be compared and contrasted, in terms of inference and task structure, in a systematic fashion. The methods used to achieve aggregation are different, for example. MARIS and SAFE are both semiautomatic. In SAFE, the user is asked to indicate line segments that should be included in a specified feature, effectively including the user only in the Aggregation needed to support Match. MARIS, however, allows the operator both to influence Aggregation, by directing line tracking algorithms to particular data items, and correct classification errors by moving lines between the various map layers. At the task level, MARIS employs the standard forward reasoning approach, while SAFE uses the structure described above. To construct reliable knowledge-level descriptions, however, requires detailed and accurate knowledge of the system(s) concerned; note that Clancey [7] had access to the source code of many of the systems he considered.

4 Comparing Structured Systems at the Knowledge Level

Some developers have sought to produce more structured knowledge-based drawing interpretation systems. Two drawing interpretation systems which comprise similar components communicating in similar ways are the mechanical drawing interpretation system of Joseph and Pridmore [13] and the utility map interpretation system of den Hartog, ten Kate and Gerbrands[15].

ANON [13] extracts 2D graphical elements from grey level images of piece-part drawings. At the heart of ANON is a hierarchically organised set of schema classes describing prototypical drawing constructs. Schema instances contain a geometrical description of the construct they represent, state variables noting the current condition of that representation and a number of C/C++ functions. The latter both manipulate schema descriptions; creating new instances, accessing and adding components, performing tests on and modifying state variables, etc. and interface to the system's low-level image analysis operations; all image analysis in ANON is performed under the control of some schema. ANON's control structure is based upon the cycle of perception model. The controlling schema both directs image analysis and interprets the result. A high-level control system is informed of this interpretation and responds by modifying the controlling schema. ANON's control system comprises rules written in an LR(1) grammar and applied by a parser generated using the Unix utility yacc.

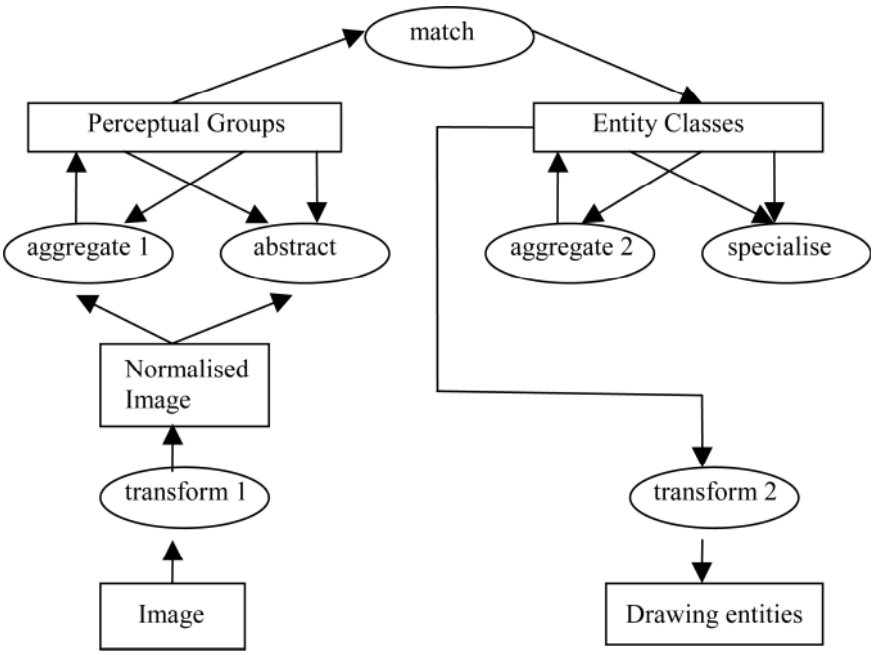

Fig.3. The inference structure of ANON

ANON's rules take two forms. The first describe relations required between constructs. Within this rule set two sub-types can be identified; both Aggregate data, (e.g. adding new segments to a broken line schema) but one also creates new, higher-level instances, bringing about an Abstract inference. Members of the second rule set perform tests on the current schema. These often Specialise the controlling schema but can also perform a Match. ANON often pursues lines of inquiry that do not lead to formation of acceptable drawing entities. A decision must be made as to whether a developing schema should be retained or discarded. This is achieved by Matching the current schema against the requirements for some drawing construct(s).

It is unfortunate to find that individual rules in ANON can perform more than one inference, but as the system was developed without any reference to knowledge engineering methodologies it is not surprising. The mapping between ANON's high level rules and KADS/CommonKADS inferences remains comparatively simple.

ANON performs a variant of heuristic classification (Fig. 3) in which Aggregation and Abstraction are iterated until a schema instance is created which can be Matched to a drawing entity class. That entity is then extended by further Aggregation and/or Specialised until no further operations are possible, at which point a specific entity description is produced. It will be noted that an initial Transform 1 inference appears, as in SAFE While ANON performs all image analysis (thresholding, etc) under schema control, the grey level image is first normalised and various local image statistics produced. The Transform 2 inference performs administrative operations on the final schema description, computing, for example, means and standard deviations of line and break lengths. ANON's task structure is difficult to specify in detail as the

pattern of calls to the Aggregate 1/Abstract and Aggregate 2/Specialise pairings varies as interpretation progresses., the process is, however, essentially forward reasoning.

The MDUS system of Dori and Liu [14] adopts a similar structure. MDUS operates on an initially unstructured vector representation of the input drawing, grouping raw vectors into higher-level constructs. At the heart of the system is a hierarchy of graphic object classes, each containing specialised object recognition algorithms based upon the same generic procedure. In MDUS, identification of the "key component" of a drawing entity triggers the creation of a hypothesis, an instance of the appropriate graphic object. That object then seeks to verify itself through application of an object-specific "extend" method that searches for further object components. When no further extension is possible the object is tested and, if credible, added to the database. Objects that fail the test are deleted. Objects are recognised class by class. As the various recognition methods are "relatively independent" of each other the order adopted is not very critical [14] though it is accepted that determining the best order in which to search for objects is a difficult problem. MDUS avoids this by allowing searches for one object class to be dynamically triggered during the search for another. A search for arrowheads might, for example, be triggered while seeking leader lines.

MDUS implements a variant of heuristic classification similar to ANON's, though it is less easy to isolate the match inference as this appears to be embedded within the class-specific code. The initial Transform (which might be better described as Aggregation) creates a vector map. A hypothesis (Perceptual Group) is then generated and extended (Aggregated). At some point top-down searches for other entities begin to appear as the class is specialised.

The map interpretation system of den Hartog et al [15] also appears very similar to ANON. It employs an object-oriented, semantic net representation of drawing entities and their interrelationships and is based around a three-component, cyclic control structure. A top-down search generation process requests a search for a specific entity in a specific area of the image, much as ANON's current schema performs a search determined by its class and state variables. In den Hartog's system the result is passed to an inconsistency detection module. In ANON, inconsistency is detected occurs by the yacc-generated parser. Both systems use the classified object returned by the search to determine the next search action, repeating the process until acceptable representations are produced or no more possibilities remain to be explored.

Despite these surface similarities the inference structures of the two systems are very different. Den Hartog et al's system maps neatly, not to heuristic classification, but to a form of incremental configuration (Fig.4). Note first that the two Decompose inferences are not necessary, as all the representations are already available in component form. The semantic net provides the "element demands and constraints" found in incremental configuration and is used to determine what should be searched for (required design elements) in what part of the drawing (grouping context). The semantic net also determines, via the grouping context, which parts of the developing interpretation need to be considered during the inconsistency detection (Assemble) phase. Only den Hartog et al's object recognition step does not fit naturally into incremental configuration, though incremental design does include the Compare inference to check that the design elements chosen are sufficient. One could consider "current design elements" to include low level geometric primitives and the Compare

inference to represent object recognition. Instead we prefer to explicitly Compare that the required entities with geometrical primitives Aggregated from the input image.

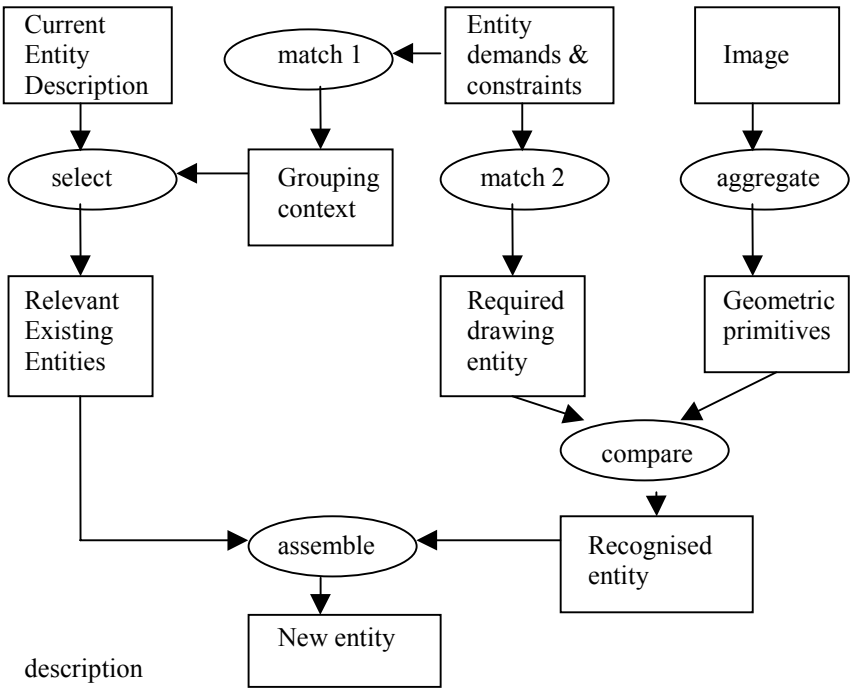

Fig. 4. The inference structure of den Hartog's map interpretation system

5 Conclusion

Reviews of various aspects of image interpretation are common in the literature and, though they often provide useful guides to those new to the field, it can be argued that most do not advance the state of their particular art. We would like to stress that the present paper is less a review than an analysis of knowledge-based line drawing understanding systems. Although much has been made of the importance of knowledge in drawing understanding, we believe that to date the field has lacked the tools required to best determine, make use of and even understand the role of the available knowledge. While almost certainly not perfect, we further believe that the tools and techniques developed under the banner of Knowledge Engineering have significant potential in this area. The analyses presented here have shown how generic task models can be used to describe, compare and distinguish between knowledge-based image interpretation systems. They have also generated first versions of models that might in time be used to inform the design of new systems. This work is ongoing; a more comprehensive knowledge-level analysis of line drawing interpretation systems is in preparation. The use of generic task models to drive the design of new

systems will be the subject of future reports. In the meantime we would urge colleagues working in knowledge-based drawing understanding systems to give some consideration to the generic task models, and particularly inference structures, underlying their systems.

References

1. B. Wielinga, A. Th. Schreiber and J. A. Breuker, "KADS: A Modeling Approach to Knowledge Engineering," Knowledge Acquisition, Vol. 5, pp. 5-53, 1992.
2. G. Schreiber, B. Wielinga and J. Breuker, (ed.), KADS: A Principled Approach to Knowledge-Based System Development. London: Academic Press Ltd., 1993.
3. D. S. W. Tansley and C. C. Hayball, Knowledge-Based Systems Analysis & Design: A KADS Developer's Handbook. Prentice Hall International (UK) Ltd., 1993.
4. G. Schreiber, et. al., Knowledge Engineering and Management: The CommonKADS Methodology. Cambridge, Mass.: MIT Press, 1999.
5. Chandresakaran, B., Generic Tasks in Knowledge-based Reasoning: High Level Building Blocks for Expert System Design, IEEE Expert, pp 23-30, Fall 1986.
6. D. Crevier, and R. Lepage, "Knowledge-Based Image Understanding Systems: A Survey," Computer Vision & Image Understanding, Vol. 67, no. 2, pp. 161-185, 1997.
7. W.J. Clancey, Heuristic Classification, Artificial Intelligence, Vol. 27, pp 215-251, 1985.
8. I. Phillips and A.K. Chabbra, Empirical Performance Evaluation of Graphics Recognition Systems, IEEE Transactions PAMI, Vol. 21, No. 9, pp 849-870, 1999.
9. J. Ton et. al.: "Knowledge-Based Segmentation of Landsat Images", IEEE Transactions on Geoscience & Remote Sensing, Vol. 29, No. 2, 1991.
10. A. Darwish, T.P. Pridmore and D. Elliman, Interpreting Aerial Images: A Knowledge-Level Perspective, Proc. ES-2001, pp 169-182, Cambridge, UK, 2001.
11. K. J. Goodson and P. H. Lewis, A Knowledge-based Line Recognition System, Pattern Recognition Letters, Vol 11, pp 295-304, 1990.
12. S. Suzuki and T. Yamada. Maris: map recognition input system. Pattern Recognition Vol. 23, pp 919-933, 1990.
13. S. H. Joseph, S. H., and T. P. Pridmore, Knowledge-directed interpretation of mechanical engineering drawings, IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 14, No. 9, pp. 928-940, 1992.
14. D.Dori and W. Liu, Automated CAD Conversion with the Machine Drawing Understanding System, IEEE Trans. Systems, Man and Cybernetics, Vol. 29, No. 4, pp 411-416, 1999.
15. J. E. den Hartog, T. K. ten Kate and J. J. Gerbrands, Knowledge-based Interpretation of Utility Maps, CVIU, Vol. 63, No. 1, pp105-117, 1996.