Reinterpretation of an existing NLG system in a Generic Generation Architecture

L. Cahill, C. Doran,* R. Evans, C. Mellish, D. Paiva, M. Reape, D. Scott, N. Tipper
Universities of Brighton and Edinburgh.
Email rags@itri.brighton.ac.uk

Abstract

The RAGS project aims to define a reference architecture for Natural Language Generation (NLG) systems. Currently the major part of this architecture consists of a set of datatype definitions for specifying the input and output formats for modules within NLG systems. In this paper we describe our efforts to reinterpret an existing NLG system in terms of these definitions. The system chosen was the Caption Generation System.

1 Introduction

The RAGS project[1] aims to define a reference architecture for natural language generation systems. Currently the major part of this architecture consists of a set of datatype definitions for specifying the input and output formats for modules within NLG systems. The intention is that such representations can be used to assist in reusability of components of NLG systems. System components that adhere to these representations, or use a format that can be translated into such representations relatively easily, can then, in principle, be substituted into other systems. Also, individual components could be developed without the need for a complete system if datasets, based on the representations, were made available.

In this paper we describe an attempt to reinterpret an existing NLG system in terms of the RAGS data definitions. The point of this exercise was to learn:

1. Whether these data structures were sufficient to describe the input and output functionality of an existing, independently developed, *applied* [2] NLG system.
2. Which aspects of the RAGS repertoire would actually be required for such a reinterpretation, which would be unnecessary and which additions to the RAGS repertoire would be motivated.
3. Whether studying the system would generate good ideas about possible reusable generation modules that could be developed.

In this exercise it was important to choose a system that had been developed by people outside the RAGS project. Equally, it was important to have sufficient clear information about the system in the available literature, and/or by means of personal contact with the developers. The system chosen was the Caption Generation System (Mittal et al., 1995; Mittal et al., 1998)[3]. This system was chosen because, as well as fulfilling the criteria above, it appeared to be a relatively simple pipeline, thus avoiding complex control issues, with individual modules performing the varied linguistic tasks that the RAGS data structures had been designed to handle.

The reinterpretation exercise took the form of coming up with an account of how the interfaces to the CGS modules corresponded to the RAGS model and reimplementing a working version of each module (apart from Text Planning and Realisation) which was tested to ensure that, given appropriate input, its output was correct (i.e. conforming to the global account) on key examples. Naturally, given the scope of this exercise, we had to gloss over some interesting implementational issues. The aim was not to produce a complete system or a system as good as CGS, but merely to demonstrate that the broad functionality of the system could be reproduced within the RAGS structures.

In this paper we first describe the RAGS data structures. We then describe the CGS system

---

* Now at the MITRE Corporation, Bedford, MA, USA, cdoran@mitre.org.

[1] This work was supported by ESPRC grants GR/L77041 (Edinburgh) and GR/L77102 (Brighton), *RAGS: Reference Architecture for Generation Systems.*

[2] See (Paiva, 1998) for a definition of *applied* in this specific context.

[3] In addition to these published sources, we were greatly helped by the developers of the system who gave us the benefit of their own expertise as well as access to the original code of the system and a technical report that included implementational details such as system traces.

followed by our reinterpretation of the system in RAGS terms. Finally we discuss the implications for RAGS of this exercise.

2 The RAGS datatypes

The RAGS project initially set out to develop a reference architecture based on the three-stage pipeline suggested by Reiter (Reiter, 1994). However, a detailed analysis of existing applied NLG systems (Cahill and Reape, 1998) suggested that such an architecture was not specific enough and not closely enough adhered to by the majority of the systems surveyed for this to be used as the basis of the architecture.

The abstract functionality of a generation system can be specified without specific reference to processing. The RAGS approach to this is to develop a **data model,** that is, to define the functional modules entirely in terms of the datatypes they manipulate and the operations they can perform on them. On top of such a model, more specific **process models** can be created in terms of constraints on the order and level of instantiation of different types of data in the data model. A 'rational reconstruction' of some pipeline model might then be produced, but other process models would also be possible.

The RAGS levels of representation are as follows[4]:

Conceptual The conceptual level of representation is defined only indirectly through an API via which a knowledge base (providing the content from which generation takes place) can be viewed as if it were defined in a simple KL-ONE (Brachman and Schmolze, 1985) like system.

Abstract Semantic Abstract semantic representations are the first level at which semantic predicates are associated with arguments. At this level, semantic predicates and roles are those used in the API to query the knowledge base and arguments are knowledge base entities.

Semantic (Concrete) semantic representations provide a complete notation for "logical forms" where there is no longer any reference to the knowledge base. The representations are based on systems such as SPL (Kasper, 1989) and DRT (Kamp and Reyle, 1993).

[4]More details can be found in (Cahill et al., 1999) and at the RAGS project web site: `http://www.itri.brighton.ac.uk/rags`.

Abstract Rhetorical Abstract Rhetorical Representations are tree structures with rhetorical relations at the internal nodes and Abstract Rhetorical trees or Abstract Semantic Representations at the leaves.

Rhetorical Abstract Rhetorical Representations are viewed as descriptions of sets of possible Rhetorical Representations. Each one may be transformed into some subset of the possible Rhetorical Representations by means of a set of permitted transformations, e.g. reversing the order of nucleus and satellite or changing the rhetorical relation to one within a permitted set.

Abstract Document Document structure defines the linear ordering of the constituents of the Rhetorical Representation with a POSITION feature, as well as two other features, TEXT-LEVEL, which takes values such as *paragraph* or *sentence*; and LAYOUT, which takes values such as *wrapped-text* and *vertical list*. It takes the form of a tree, usually, but not necessarily, isomorphic to the Rhetorical Representation and linked to it, but with these three features at the nodes instead of rhetorical relations.

Abstract Syntactic Abstract Syntactic Representations capture high-level aspects of syntactic structure in terms of notions such as lexical head, specifiers, modifiers and complements. This level of representation is compatible with approaches such as LFG f-structure, HPSG and Meteer's Text Structure.

3 Partial and Mixed Representations

For all of the RAGS levels **partial** representations are possible. Without this, it is not possible for a module to pass any result to another until that result is completely determined, and this would impose an unwanted bias towards simple pipeline architectures into the model. There are many cases in NLG where a representation is built collaboratively by several modules. For instance, many systems have a referring expression generation module whose task is to complete a semantic representation which lacks those structures which will be realised as NPs. Such a functionality cannot be described unless partially complete semantic representations can be communicated.

In addition, **mixed** representations are possible, where (possibly partial) representations at several levels are combined with explicit links between the elements. Many NLG modules have to be sensitive to a number of levels at once (consider, for instance, aggregation, referring expression generation and lexicalisation, all of which need to take into account rhetorical, semantic and syntactic constraints). The input to most reusable realisation systems is also best viewed as a mixture of semantic and abstract syntactic information.

The extra flexibility of having partial and mixed representations turned out to be vital in the reconstruction of the CGS system (Mellish et al., 2000).

4 The CGS system

The Caption Generation System (CGS) generates explanatory captions of graphical presentations (2-D charts and graphs). Its architecture is a pipeline with several modules, shown in the left hand part of Figure 1. An example of a diagram and its accompanying text are given in Figure 2. The propositions are numbered for ease of reference throughout the paper.

The input to CGS is a picture representation (graphical elements and its mapping from the data set) generated by SAGE plus its complexity metric. The text planning module (Moore and Paris (1993)) plans an explanation in terms of high level discourse goals. The output of the planner is a partially ordered plan with speech-acts as leaves.

The ordering module receives as input the discourse plan with links specifying the ordering relations between sub-trees and specifies an order for them based on heuristics such as that the description should be done from left to right in the visual space.

The aggregation module "only conjoins pairs of contiguous propositions about the same grapheme type[5] in the same space" (Mittal et al., 1999) and inserts cue phrases compatible with the propositions (e.g., "whereas" for contrastive ones). The internal order of the sentence constituents is determined by the centering module using an extension of the centering theory of Grosz and colleagues (Grosz et al., 1995).

The referring expression module uses Dale and Reiter's (Dale and Reiter, 1995) algorithm to construct the set of attributes that can uniquely identify a referent. There are two situations where the text planning module helps specifically in the generation of referring expressions: (1) when the complexity for expressing a graphic demands an example and

[5]"*Graphemes* are the basic building blocks for constructing pictures. Marks, text, lines and bars are some of the different grapheme classes available in SAGE." (Mittal et al., 1999).

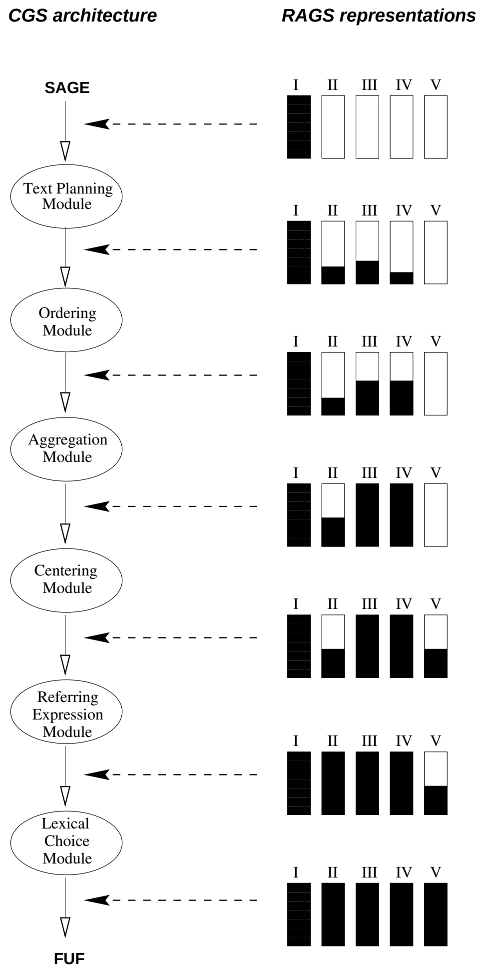

Figure 1: A RAGS view of the CGS system. The labels for the RAGS representations refer to the following: I = conceptual; II = semantic; III = rhetorical; IV = document; V = syntactic.

it signals this both to SAGE (for highlighting the corresponding grapheme) and to the rest of the text generation modules; and (2) when in a specific situation the referring algorithm would need several interactions for detecting that an entity is unique in a certain visual space and the planning could detect it in the construction of the description of this space. When this occurs, the text planner "circumvents the problem for the referring expression module at the planning stage itself, processing the speech-acts appropriately to avoid this situation completely".

After lexicalisation, which adds lexeme and major category information, the resulting functional descriptions are passed to the FUF/SURGE realiser that generates texts like the caption of Figure 2.

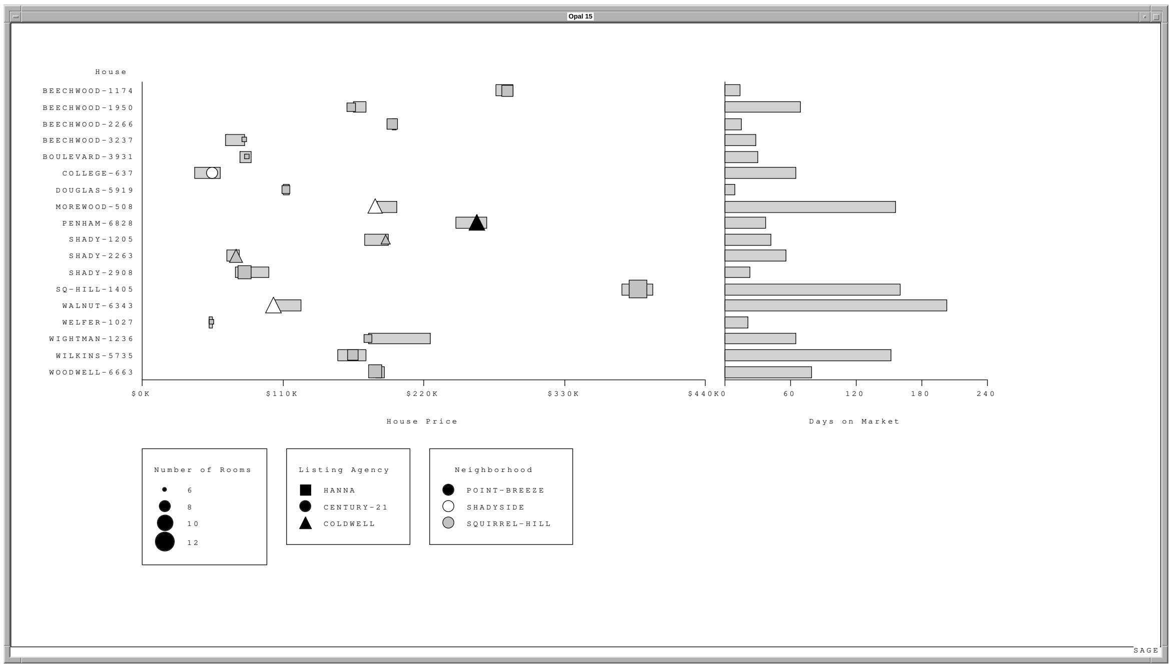

Figure 2: (1) These two charts present information about house sales from data-set ts-1740. (2) In the two charts, the y-axis indicates the houses. (7) In the first chart, the left edge of the bar shows the house's selling price whereas (8) the right edge shows the asking price. (3) The horizontal position of the mark shows the agency estimate. (4) The color shows the neighbourhood and (5) shape shows the listing agency. (6) Size shows the number of rooms. (9) The second chart shows the number of days on the market.

5 Reinterpretation of CGS in RAGS

Our reinterpretation of the CGS system defines the interfaces between the modules of CGS in terms of the RAGS data structures discussed above. In this section we discuss the input and output interfaces for each CGS module in turn as well as any problems we encountered in mapping the structures into RAGS structures. Figure 1 shows the incremental build-up of the RAGS data levels across the pipeline. Here we have collapsed the Abstract Rhetorical and Rhetorical and the Abstract Semantic and Semantic. It is interesting to note that the build up of levels of representation does not tend to correspond exactly with module boundaries.

One of the major issues we faced in our reinterpretation was where to produce representations (or partial representations) whose emergence was not defined clearly in the descriptions of CGS. For instance, many decisions about document structure are made only implicitly by the system. In most cases we have opted to produce all types of representations at the earliest point where they can conceivably have any content. This means, for instance, that our reimplementation assumes an (unimplemented) text planner which produces an Abstract Rhetorical Representation with Abstract Semantic leaves and an Abstract Document Representation.

Text Planner The input to the Longbow text planner discussed in section 4 above is a representation of a picture in SAGE format (which has been annotated to indicate the types of complexity of each grapheme) together with a goal, which can typically be interpreted as "describe". It outputs an essentially flat sequence of plan operators, each of which corresponds in the output text to a speech act. In our reinterpretation, we have assumed that this flat structure needs to be translated into an Abstract Rhetorical Representation with (at least) minimal structure. Such a structure is implicit in the plan steps, and our interpretation of the rhetorical structure for the example text corresponds closely to that of the post-processing trace produced by CGS.

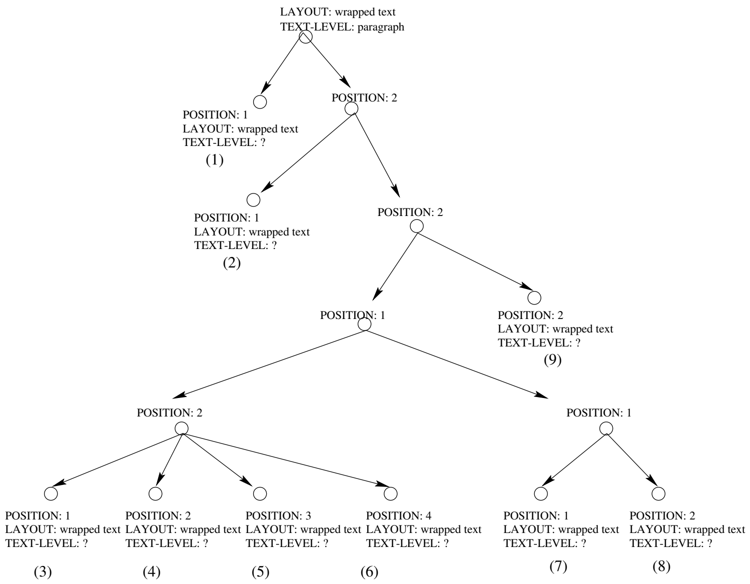

Figure 3: Initial Document Structure

However, we are still not entirely sure about where exactly CGS creates this structure, so we have imposed it at the very beginning, onto the output of the text planner.

Already at this stage it is necessary to make use of mixed RAGS representations. As well as this Abstract Rhetorical Representation, the text planner has to produce an Abstract Document Representation, linked to the Abstract Rhetorical Representation. This is already partially ordered – although the exact value of POSITION features cannot be specified at this stage, the document tree is constructed so that propositions are already grouped together. In addition, we make explicit certain default information that the CGS leaves implicit at this stage, namely, that the LAYOUT feature is always *wrapped text* and that the TEXT-LEVEL feature of the top node is always *paragraph*.

Ordering The ordering module takes the Abstract Document Representation and the Abstract Rhetorical Representation as input and outputs an Abstract Document Representation with the POSITION feature's value filled for all the nodes. That is, it fixes the linear order of the final output of the speech acts. In our example, the ordering is changed so that steps 7 and 8 are promoted to appear before 3, 4, 5 and 6. The resulting structure is shown in figure 3[6].

Aggregation Although aggregation might seem like a self-contained process within NLG, in practice it can make changes at a number of levels of representation and indeed it may be the last operation that has an effect on several levels. The aggregation module in our reinterpretation thus has the final responsibility to convert an Abstract Rhetorical Representation with Abstract Semantic Representation leaves into a Rhetorical Representation with Semantic Representation leaves. The new Rhetorical Representation may be different from before as a result of speech acts being aggregated but whether different or not, it can now be considered final as it will no longer be changed by the system. The resulting Semantic Representations are no longer Abstract because further structure may have been determined for arguments to predicates. On the other hand, referring expressions have not yet been generated and so the (Concrete) Semantic Representations cannot be complete. The reconstruction creates partial Semantic Representations with "holes" where the referring expressions (Semantic Representations) will be inserted. These "holes" are linked back to the knowledge base entities that they correspond to.

Because Aggregation affects text levels, it also affects the Abstract Document Representation, which has its TEXT-LEVEL feature's values all filled at this

[6] In this and the following diagrams, *objects* are represented by circles with (labelled) arrows indicating the relations between them. Dashed arrows indicate links between different levels of representation.

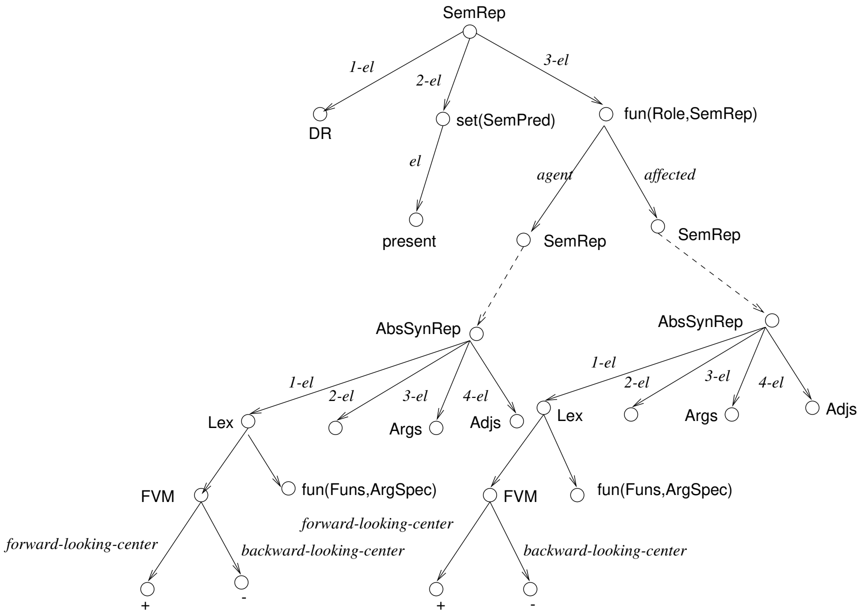

Figure 4: Syntactic representations constructed by Centering

point. It may also need to change the structure of the Abstract Document Representation, for instance, adding in a node for a sentence above two, now aggregated, clause nodes.

Centering Because Centering comes before Referring Expression generation and Realisation, all it can do is establish constraints that must be heeded by the later modules. At one stage, it seemed as if this required communicating a kind of information that was not covered by the RAGS datatypes. However, the fact that an NP corresponds (or not) to a center of some kind can be regarded as a kind of abstract syntactic information. The reconstruction therefore has the centering module building a partial (unconnected) Abstract Syntactic representation for each Semantic Representation that will be realised as an NP, inserting a feature that specifies whether it constitutes a forward- or backward-facing center, approximately following Grosz et al (Grosz et al., 1995). This information is used to determine whether active or passive voice will be used. An example of such a partial Abstract Syntactic Representation is given in Figure 4.

Referring Expression In our reconstruction of the CGS system, we have deviated from reproducing the exact functionality for the referring expression module and part of the lexical choice module. In the CGS system, the referring expression module computes association lists which can be used by the lexical choice module to construct referring expressions suitable for realisation. In our reconstruction, however, the referring expression module directly computes the Semantic Representations of referring expressions.

We believe that this is a good example of a case where developing a system with the RAGS data structures in mind simplifies the task. There are undoubtedly many different ways in which the same results could be achieved, and there are many (linguistic, engineering etc.) reasons for choosing one rather than another. Our particular choice is driven by the desire for conceptual simplicity, rather than any strictly linguistic or computational motivations. We considered for each module which RAGS level(s) it contributed to and then implemented it to manipulate that (or those) level(s). In this case, that meant a much more conceptually simple module which just adds information to the Semantic Representations.

Lexical Choice In CGS, this module performs a range of tasks, including what we might call the later stages of referring expression generation and lexical choice, before converting the plan leaves into FDs (Functional Descriptions), which serve as the input to the FUF/SURGE module. In the reconstruction, on the other hand, referring expressions have already been computed and the Rhetorical Representation, with its now complete Semantic Representations, needs to be "lexicalised" and

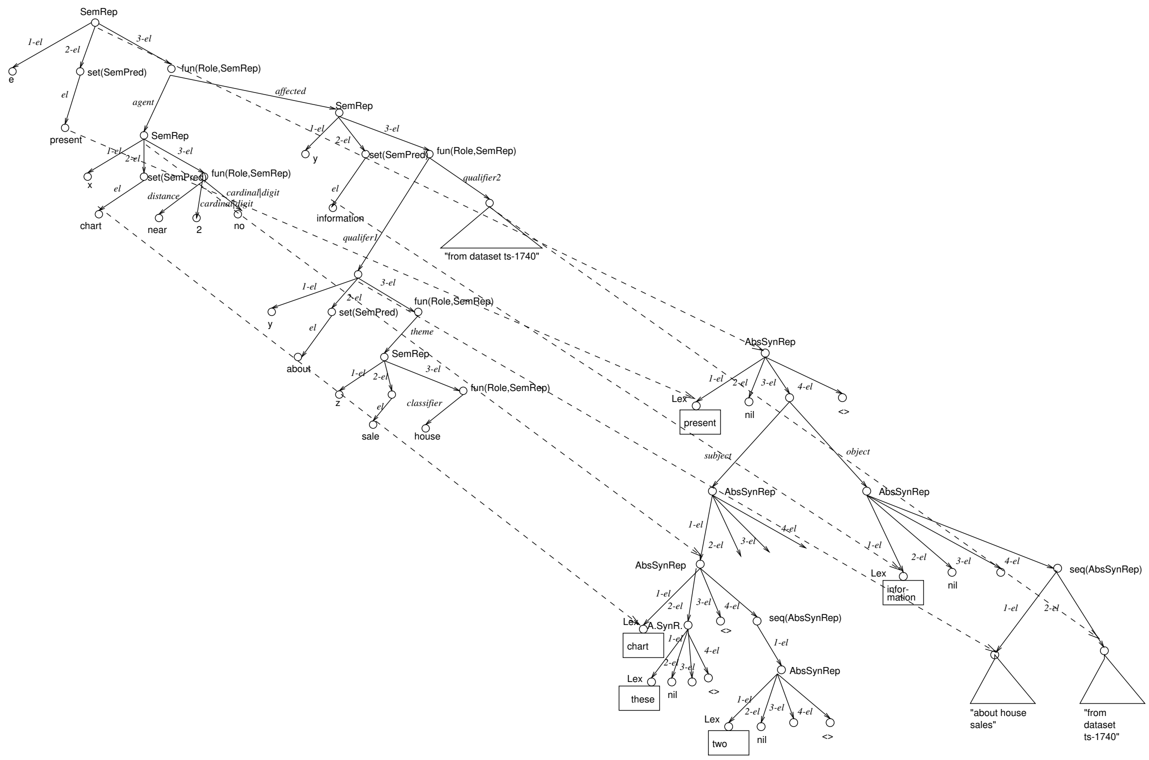

Figure 5: Combined Semantic and Abstract Syntactic Representation

translated into FUF/SURGE format. Lexicalisation in our terms involves adding the lexeme and major category information to the Abstract Syntactic Representations for the semantic predicates in each Semantic Representation. The FUF/SURGE input format was regarded as a combination of Semantic and Abstract Syntactic information, and this can easily be produced from the RAGS representations. The combined Semantic and Abstract Syntactic Representations for the plan step "These two charts present information about house sales from data set ts-1740" is shown in Figure 5. The boxes indicate suppressed subgraphs of the lexemes corresponding to the word in the boxes and triangles indicate suppressed subgraphs of the two adjuncts.

6 Conclusions

The reconstruction of CGS has taken the form of working out in detail the RAGS representations passed between modules at each stage for a set of key examples and reimplementing the modules (apart from the Planner and Realiser) in a way that correctly reproduces these representations. The actual implementation used an incrementally growing data store for the RAGS representations which the modules accessed in turn, though the passing of data could also have been achieved in other ways.

The fact that the reconstruction has been successful indicates that the RAGS architecture is broadly adequate to redescribe this NLG system:

- No changes to the existing levels of representation were needed, though it was necessary to make extensive use of partial and mixed representations.
- No new levels of representation needed to be introduced to capture the inter-module communication of the system.
- All of the levels of representation apart from the Conceptual level were used significantly in the reconstruction.

In some ways, it is unfortunate that none of the inter-module interfaces of CGS turned out to use a single level of RAGS representation. Given the motivation for partial and mixed representations above, however, this did not really come as a surprise. It may well be that any really useful reusable modules for NLG will have to have this complexity.

In spite of the successful testing of the RAGS data model, some difficulties were encountered:

- It was difficult to determine the exact nature of the representations produced by the Planner, though in the end we were able to develop a system to automatically translate these into a format we could deal with.
- Although the theoretical model of CGS has a simple modular structure, in practice the modules are very tightly integrated and making the exact interfaces explicit was not always easy.
- Referring expression generation requires further access to the "knowledge base" holding information about the graphic to be produced. This knowledge was only available via interactions with SAGE, and so it was not possible to determine whether the RAGS view of Conceptual Representations was applicable. Our own implementation of referring expression generation had to work around this problem in a non-portable way.
- It became clear that there are many housekeeping tasks that an NLG system must perform following Lexical Choice in order for the final Semantic and Abstract Syntactic Representations to be appropriate for direct input to a realisation system such as FUF.
- The fact that the system was driving FUF/SURGE seems to have had a significant effect on the internal representations used by CGS. The reconstruction echoed this and as a result may not be as general as could be desired.
- Even though CGS only performs simple types of Aggregation, it is clear that this is a critical module for determining the final form of several levels of representation.

The division of CGS into modules is different from that used in any NLG systems we have previously worked on and so has been a useful stimulus to think about ways in which reusable modules can be designed. We envisage reusing at least the reimplementation of the Centering module in our further work.

References

R. Brachman and J. Schmolze. 1985. An overview of the KL-ONE knowledge representation system. *Cognitive Science*, 9:171–216.

Lynne Cahill and Mike Reape. 1998. Component tasks in applied NLG systems. Technical Report ITRI-99-05, ITRI, University of Brighton. obtainable at http://www.itri.brighton.ac.uk/projects/rags/.

Lynne Cahill, Christy Doran, Roger Evans, Chris Mellish, Daniel Paiva, Mike Reape, Donia Scott, and Neil Tipper. 1999. In Search of a Reference Architecture for NLG Systems. In *Proceedings of the 7th European Workshop on Natural Language Generation*, pages 77–85, Toulouse.

Robert Dale and Ehud Reiter. 1995. Computational interpretations of the Gricean maxims in the generation of referring expressions. *Cognitive Science*, 18:233–263.

B.J. Grosz, A.K. Joshi, and S. Weinstein. 1995. Centering: a framework for modelling the local coherence of discourse. *Computational Linguistics*, 21(2):203–226.

H. Kamp and U. Reyle. 1993. *From discourse to logic: Introduction to model theoretic semantics of natural language, formal logic and discourse representation theory*. Kluwer, Dordrecht; London.

R. T. Kasper. 1989. A flexible interface for linking applications to penman's sentence generator. In *Proceedings of the DARPA Speech and Natural Language Workshop*, Philadelphia.

C. Mellish, R. Evans, L. Cahill, C. Doran, D. Paiva, M. Reape, D. Scott, and N. Tipper. 2000. A representation for complex and evolving data dependencies in generation. In *Proceedings of the Applied Natural Language Processing (ANLP-NAACL2000) Conference*, Seattle.

V. O. Mittal, S. Roth, J. D. Moore, J. Mattis, and G. Carenini. 1995. Generating explanatory captions for information graphics. In *Proceedings of the 15th International Joint Conference on Artificial Intelligence (IJCAI'95)*, pages 1276–1283, Montreal, Canada, August.

V. O. Mittal, J. D. Moore, G. Carenini, and S. Roth. 1998. Describing complex charts in natural language: A caption generation system. *Computational Linguistics*, 24(3):431–468.

Daniel Paiva. 1998. A survey of applied natural language generation systems. Technical Report ITRI-98-03, Information Technology Research Institute (ITRI), University of Brighton. Available at http://www.itri.brighton.ac.uk/techreports.

Ehud Reiter. 1994. Has a consensus NL generation architecture appeared and is it psycholinguistically plausible? In *Proceedings of the Seventh International Workshop on Natural Language Generation*, pages 163–170, Kennebunkport, Maine.

Acknowledgements

We would like to thank the numerous people who have helped us in this work. The developers of CGS, especially Giuseppe Carenini and Vibhu Mittal; the RAGS consultants and other colleagues at Brighton and Edinburgh, who have contributed greatly to our development of the representations; and finally to the anonymous reviewers of this paper.