Achieving Theory-Neutrality in Reference Architectures for NLP: To what extent is it possible/desirable?

Lynne Cahill*, Christy Doran*, Roger Evans*, Chris Mellish†, Daniel Paiva*, Mike Reape† and Donia Scott*

*ITRI, University of Brighton
Email: *firstname.lastname*@itri.bton.ac.uk

† Division of Informatics, University of Edinburgh
Email: {chrism, reape}@dai.ed.ac.uk

April, 1999

Abstract

The recent trend towards the development of reference architectures in NLP has reached deep into NLU (Natural Language Understanding), with the GATE, Tipster and MUC enterprises, while largely maintaining neutrality with regard to linguistic theories. In NLG (Natural Language Generation), however, the move towards reference architectures has only just begun, and is likely to be harder to achieve than in NLU, largely because of the lack of a standard input. In this paper we discuss how the RAGS project aims to develop a reference architecture for NLG, and how and to what extent we expect to achieve theory neutrality.

1 Reference Architectures and Theory Neutrality

There are a number of possible goals associated with the process of designing reference architectures:

- Reuse of existing ideas, even software
- General provision of standard interfaces and datasets
- Basis for evaluation, specialisation
- Support for application and research system development
- Stimulation of new developments (e.g. by highlighting major problems or gaps)

Additional benefits can include:

- Rational reconstruction of existing practice
- Motivated improvement on current practice

Linguistic theories abound in NLP. The level of theory neutrality of an architecture affects the goals that can be achieved by it. Existing work on general architectures for NLU has shied away from theoretical commitment, in favour of general infrastructure support. For instance, the Tipster and GATE architectures [Gaizauskas et al (1996)] are deliberately designed to allow arbitrary theories that can be expressed in terms of annotations on text spans. The MUC conferences [Chinchor (1998)] have turned away from more "semantics-related" evaluation tasks which require theory commitment. However, more theoretical commitment seems to be inevitable when the community addresses the evaluation and coordination of complete NLU systems. It is also necessary for natural language generation (NLG), where there is no "given" input structure to anchor representations to.

The question then is how to manage this, as an architecture that embodies a very specific theory is unlikely to be adopted by the community in general.

2 The RAGS Project

The RAGS project is developing a reference architecture for NLG systems. Its goals are (in order):

1. Support for applications-oriented system development
2. General provision of standard interfaces and datasets
3. Reuse of existing ideas and software
4. Basis for evaluation, specialisation

The inspiration for the project came from Reiter's [Reiter (1994)] examination of five "complete" NLG systems, which concluded that there was an emerging consensus pipeline architecture, as we have restated in Figure 1.

In RAGS we have extended Reiter's study, examining a further 20 systems to identify Reiter's components. The model was found to be generally valid. For instance, Figure 2 shows how the modules of two existing NLG systems correspond to Reiter's components.

In order to determine what the task of each processing component should be in the Reiter model, we considered the linguistic tasks of lexicalisation, aggregation, rhetorical structuring, referring expression generation, ordering, segmentation and centering/salience/theme assignment. With the exception of centering/salience/theme, most of these tasks occur in most systems. There is a clear distinction between those systems which accept input directly from the user and those which accept input from an existing knowledge base. However, the relation between component linguistic tasks and the pipeline model is unclear. Figure 3 shows the results of our analysis of where the tasks took place in 10 systems that took their input from a knowledge base.

As the table shows, there is an uneasy fit between the linguistic tasks and the three "Reiter modules". Processing of some tasks in some systems is not local to one module, but spread across different parts of the pipeline. Even where the tasks are pipelined, the ordering constraints imposed by them are only partially constraining.

3 Towards a Reference Architecture

There are several possible accounts for our difficulty in clearly locating the processing tasks within the pipeline. It may be that particular theories may require that tasks are done in a certain order (e.g. Meaning Text Theory-based systems), or this may just be an arbitrary property of an implementation. Moreover, it can be difficult to get a clear view of what tasks are being performed where, as system descriptions differ greatly in their use of terminology. Sometimes similar results can be achieved in more than one way, using operations at different levels (e.g., with aggregation). Sometimes what seems to be an atomic task may have components which can be distributed through the architecture (e.g., early vs. late lexicalisation [Cahill (1998)]).

To reconcile these two non-isomorphic conceptualizations we propose to develop a reference architecture based on two parts:

An abstract data model which specifies the types of data manipulated by an NLG system and what operations are *possible* on them.

More specific architectures constructed by adding further

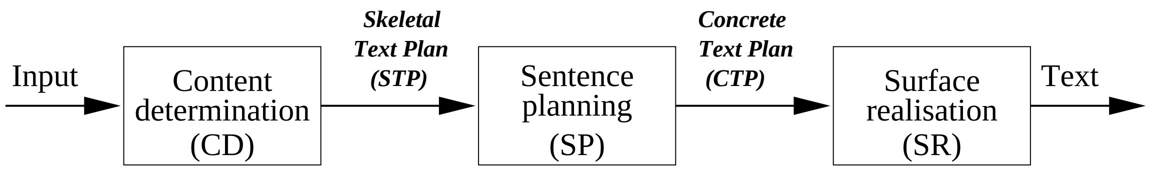

Figure 1: Simplified Generation Architecture

ModEx	CD	SP	SR
Text planner	•		
Sentence planner		•	
Realizer			•
Formatter			•

Healthdoc	CD	SP	SR
Authoring tool	•		
Selection of content	•		
Sentence repair		•	
Realization and formatting			•

Figure 2: Evidence for Reiter's Components

constraints to the ordering of operations.

The intention is that someone implementing an operation in the abstract data model can guarantee that it will be appropriate in all instantiations of the architecture. Researchers are free to find interesting ways of scheduling these operations in a complete system. The more specific architectures exist as guidance and to facilitate sharing of datasets that span multiple levels.

For the data model, we are considering five main levels of representation: **Conceptual**, **Rhetorical**, **Semantic**, **Syntactic**, and **Document**. These types may have subtypes (e.g. lexicalised structures may be a subset of syntactic structures) and may be further specified in terms either of a concrete representation or in terms of access functions that they should support. Operations are defined in terms of the types of their inputs and outputs. For instance, perhaps, Lexicalisation receives as input (at least) a set of semantic structures and returns a set of lexicalised abstract syntactic structures. The input/output requirements of operations impose some limits on how these can be ordered, but the intention is that the data model should reflect fairly uncontentious minimal constraints. It should be possible to derive most existing architectures by further specialising the data model and imposing extra ordering constraints. For instance, most systems do not clearly make the conceptual/semantic distinction. For these systems one simply assumes additionally that conceptual structures and semantic structures are one and the same thing.

The next step is to define more specific architectures. This will be done in terms of the skeletal and complete text plan (STP and CTP) interfaces, by specifying what information definitely *must* be in each interface, what information definitely *must not* be in each interface and what information *can optionally* be in each interface. Any pair of STP/CTP definitions constrains the general architecture to be a pipeline. Some very prescriptive definitions will yield very conservative pipelines that correspond closely to one or more existing systems. Some very loose definitions will allow more flexible systems, at the expense of making it harder to implement conformant processing components. The project also has to consider to what extent it wants to capture what existing systems do and to what extent it might want to "rationally reconstruct" those systems, allowing a foundation for future development.

4 Conclusions

The development of a reference architecture for NLG is constrained in different

	Lex	Agg	Rhet	Ref	Ord	Seg	Cent/Sal
Ana	SR	SR	CD	SR	CD	SR	?
PlanDoc	SP/SR	CD/SP(?)	CD(?)	CD(?)	SR(?)	CD	CD
FoG	SP/SR	SP	CD	SR	CD	CD	X
Gossip	SP/SR	?	CD(?)	SR	CD	CD(?)	CD/SP/SR
LFS	SP/SR	CD(?)	CD	CD	CD	CD	CD/SP/SR
CGS	SR	CD	CD	SP	CD	CD(?)	CD/SP
PostGraphe	CD	CD(?)	CD	SR	CD	CD	?
Komet	SR	(SR)	CD/SP	CD/SP	CD/SP	CD/SP	CD/SP
AlethGen	SR/SP	SP	SP	SP	SP	(CD)	X
Proverb	SP	SP	CD	SP	CD/SP	CD	X

Figure 3: Location of Processing Tasks

ways to that for NLU. As we have discussed, the issue of theory-neutrality is one which affects many decisions about the nature of a reference architecture, perhaps more so for NLG than for NLU. However, in order to be widely applicable across the NLG community, at least a degree of theory neutrality is clearly desirable, if not essential.

Theory-neutrality can probably be achieved in RAGS by separating the linguistic tasks from the architecture. However to make the interfaces concrete/usable we may have to introduce some theory-bound constraints. The data model can perhaps be seen as a generalisation of the GATE "text spans" backbone (both abstract away from restrictive ordering principles). Alternatively it can be seen as encapsulating the sorts of agreements that people using GATE have to make in order to interface modules.

We therefore propose to develop an architecture which maintains virtually total theory neutrality in its most basic form, but to which a variety of constraints may be added to impose theoretical limits on that archtiecture.

5 Acknowledgements

The research described here is the joint work of the RAGS team. The RAGS project is supported by EPSRC grants GR/L77041 to Chris Mellish and GR/L77102 to Donia Scott. We would like to thank our colleagues at the ITRI and Edinburgh for helpful discussions on various aspects of the work and our anonymous referees for their comments on the paper. Further details of the RAGS project can be found at **http://www.itri.bton.ac.uk/projects/rags**.

References

[Cahill (1998)] Lynne Cahill, "Lexicalisation in Applied NLG Systems", ITRI Technical report, ITRI-99-04, available at **http://www.itri.bton.ac.uk/projects/rags**.

[Chinchor (1998)] Nancy Chinchor, "Overview of MUC-7" in *Proceedings of MUC-7*, at **http://www.muc.saic.com/**.

[Gaizauskas et al (1996)] Robert Gaizauskas, Hamish Cunningham, Yorick Wilks, Peter Rodgers and Kevin Humphreys, "GATE: An Environment to Support Research and Development in Natural Language Engineering", *Procs of the 8th IEEE International Conference on Tools with Artificial Intelligence*, Toulouse, France, 1996.

[Reiter (1994)] Ehud Reiter, "Has a Consensus NL Generation Architecture Appeared, and is it Psycholinguistically Plausible?", *Procs of the Seventh International Workshop on Natural Language Generation*, Kennebunkport, Maine, 1994.