An architecture for bilingual and bidirectional NLP

Alistair Knott, Ian Bayard, Samson de Jager and Nick Wright
Dept of Computer Science
University of Otago*

Abstract

This paper describes Te Kaitito, a system for natural language processing in English and Māori. The paper focusses on the high-level architecture of the system, and the design decisions which motivate this architecture. An example of an interaction with the system is also given.

1 Introduction

This paper describes Te Kaitito:[1] a collection of natural language processing (NLP) resources for English and Māori. The system is intended to fulfil several functions. Firstly, it is intended to serve as a platform for developing computational models of syntax, semantics and discourse, and particularly as a training ground for students working in these areas. Secondly, it is intended to be the foundation for useful natural-language-processing applications, such as a natural language front-end for a database, a simple sentence translator, and a language-teaching tool. Finally, it is intended to act as a shop window for computational linguistics research in New Zealand. A focus on the Māori language is particularly important in this regard; New Zealand is a bilingual country, and it is important to develop natural language processing resources for the indigenous language as part of the effort to ensure its survival.

We will begin in Section 2 by outlining the theoretical criteria which we are attempting to satisfy in the design of the system. In Section 3 we describe the system's various different components. Section 4 gives an example of a dialogue with the system.

2 Motivations for Te Kaitito's architecture

A great deal of effort has recently been focussed on developing standard architectures for NLP—see for instance the GATE architecture (Cunningham *et al.*, 2002) and the LT-XML toolkit for corpus processing (Thompson *et al.*, 1997). These projects both concentrate on standards for relatively robust and low-level NLP tasks, such as corpus annotation, tokenising and part-of-speech tagging. Our project concentrates on higher-level linguistic tasks: parsing, semantic interpretation, dialogue and text generation. In these areas, it is still arguably premature to develop precise standards. Decisions about a suitable architecture are instead grounded in a mixture of software engineering principles (Section 2.1) and considerations specific to the kind of applications we have in mind (Sections 2.3—2.4).

2.1 Modularity

Te Kaitito is designed as a collection of NLP resources to support a family of related

*This work was supported by University of Otago Research Grant MFHB10, and by the NZ Foundation for Research in Science & Technology grant UOOX02.

[1] *Te Kaitito* is Māori for 'the composer', or 'the improviser'.

NLP applications. We envisage four applications: sentence translation (via semantics), mixed-initiative dialogue, computer-assisted language learning and dynamic hypertext (Dale *et al.*, 1998). In keeping with general software engineering principles, the system is built from a set of modular components. At a high level, there is a good consensus about the modules into which an NLP system should be decomposed: firstly there are **declarative** modules such as grammars, lexicons and ontologies; secondly there are **procedural** modules such as tokenisers, part-of-speech taggers, sentence parsers and interpreters; finally there are **interface** modules such as web or socket-based interfaces. The declarative modules in our system are grammars, lexicons and morphological rules. The procedural modules are a sentence interpreter, a sentence generator, a presupposition-resolution module, a dialogue engine, a content-selection module, a text-planner and a sentence planner. There are currently two interface modules: one for a web-based sentence translator, and one for a web-based dialogue system. These modules will be described in Section 3.

2.2 Bidirectionality

A key element of the design of Te Kaitito is that it should support *bidirectional* or reversible processing, in which the same declarative resources are used by the processes which generate language and by those which interpret it. The text generation research community has not been overly concerned with the issue of bidirectionality. Systems in which the primary research interest is in generation frequently only perform generation. Systems which perform both generation and interpretation (e.g. Shieber, 1988; Van Noord, 1993) are typically built for use in machine translation applications, where many components of the generation process are simply not required. The generation module of a machine translation system only needs to be concerned with the linguistic realisation of a sentence-sized semantic message: there is no need to consider how and why this message is itself generated, and how the message in question is integrated into a larger discourse being produced. However, there are good reasons for wanting a full-scale generation system to support sentence interpretation too. Our reasons both stem from the complexity of the knowledge bases required to support a generation system.

Firstly, in any environment in which text is being automatically generated, it would be useful to include a facility for question-answering. A generation system requires as input a knowledge base of facts in some computer-tractable representation. Given this fact, the extra effort of building a system that takes a query for this representation and generates a response should be relatively small. Likewise, a generation system requires a grammar, and a compositional mapping between grammatical structures and the semantic structures of the database representation. The extra effort of ensuring that the grammar is bidirectional should again be relatively small. On the other hand, the benefits of having a generation system which responds to natural language queries seem quite considerable. A generation system has to be able to respond flexibly to user input, but without sentence interpretation this input is typically very constrained, amounting in many cases simply to a fixed set of hyperlinks or menu options. In such cases, the poverty of the interface is often a limiting factor in the performance of the generation system, especially one which is designed to handle a sophisticated semantic representation at input.

Secondly, the performance of a generation system is also currently limited by the quality of the methods available for authoring its knowledge base. It is well known that knowledge base authoring is a bottleneck for current generation systems (see e.g. Paris, 2001). If it were possible to author a knowledge base by entering natural language sentences, this would certainly be a useful facility. Clearly, a constrained interface of some kind would be necessary to overcome problems relating to the interpretation of free text; however, there has already been a certain amount of work in this area (e.g. Power *et al.*, 1998; Piwek *et al.*, 1999).

If a system is going to support both generation and interpretation, we suggest it makes sense to use a bidirectional grammar, rather than developing specialised and independent generation and interpretation modules. See Neumann (1994) for a good summary of the advantages of bidirectional systems.

2.3 Multilinguality

From the point of view of effort versus reward, it makes sense for generation systems to target applications in which documents are to be produced in several languages (see e.g. Reiter and Dale, 2000). Given this fact, it makes sense to include a multilingual capability from the outset when a system is being developed, to avoid building language-specific assumptions into its design. Moreover, to ensure a good degree of language independence, it makes sense to aim for coverage of a set of languages which are not closely related to one another. While these are certainly not the main reasons why we are focussing on English and Māori, it is useful from this perspective that they are entirely unrelated languages.

A second decision we took was to incorporate our coverage of English and Māori into a single declarative grammar. The reason for this is partly theoretical parsimony and partly practical efficiency. From a theoretical perspective, we are interested in capturing the linguistic constraints which the languages share. From a practical perspective, we want to make it easy to add new languages which are syntactically similar to Māori (in particular Tongan and Samoan), and therefore want to develop from the outset a framework in which the similarities between two languages can be made explicit.

2.4 Semantics

Work in the semantics of natural language is not made use of as much as it could be in natural language generation. The semantics of noun phrases is an interesting case in point. Generation theorists tend to think of noun phrases as referring to objects in the world. Semanticists prefer to think of NPs in set-theoretic terms, within a general framework in which sentences are analysed as tripartite quantification structures featuring a bound variable, a 'restrictor set' and a 'nuclear scope set' (Barwise and Cooper, 1981; Heim, 1982). The determiner of a subject noun phrase introduces the bound variable and an appropriate quantification operator, the subject N-bar contributes the restrictor set, and the VP contributes the nuclear scope set. This approach seems madly complex for simple sentences, but pays off in the analysis of quantified sentences. There is a great deal of work on the generation of noun phrases (see e.g. Dale, 1988; Horacek, 1996), but the models proposed are not typically expressed in this set-theoretic vocabulary; and it is probably no coincidence that (with the exceptions of Creaney, 1996 and Power, 1999) there has not been much work on the generation of quantified sentences.

There are other topics in semantics which have likewise received little attention from generation researchers—for instance, there are few generation systems that explore the topic of presuppositions in great detail. As a general research focus, we are interested in exploring whether there are any good reasons for using analyses taken from semantic theory within a text generation system.

3 The modules in Te Kaitito

Figure 1 details the architecture of Te Kaitito. Boxes in bold denote modules of the system; rounded boxes are procedural modules and square boxes are declarative ones. (Interface modules are not shown.)

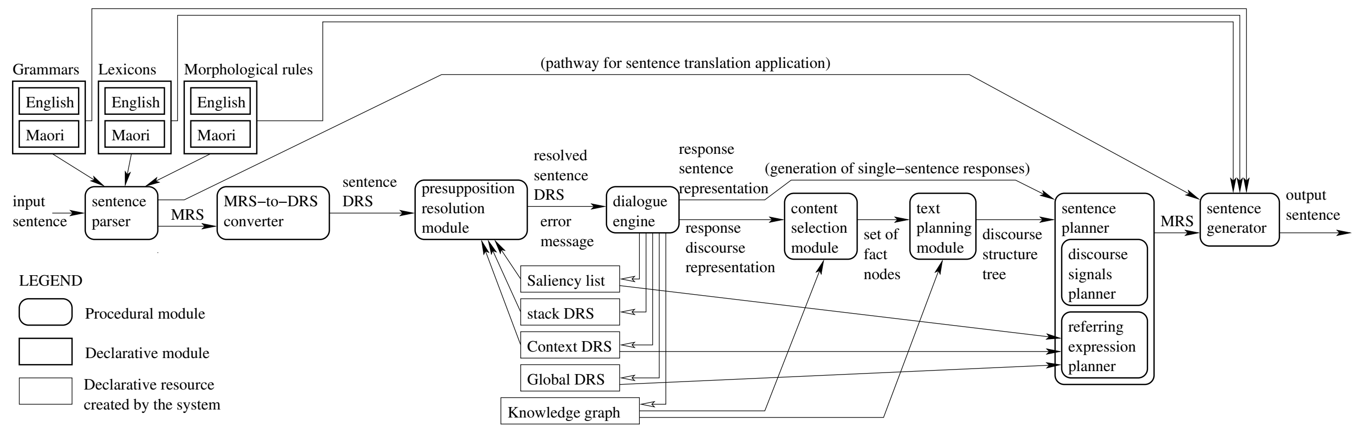

Figure 1: Architecture of Te Kaitito

3.1 Overview of pathways from input sentence to response

Three different pathways from an input sentence to an output response are shown. The simplest pathway is used in the machine translation application. An input sentence is sent to the parser, which consults grammatical, morphological and lexical resources to produce a semantic representation in a language called Minimal Recursion Semantics or **MRS** (Copestake *et al.*, 2001). This representation is sent directly to the generator, which uses the same declarative resources to produce a set of alternative sentences expressing this MRS. Section 3.2 describes the parser and the generator, and the grammatical and syntactic representations they use.

The other two pathways involve a dialogue-based interaction with the user. A sentence typed in by the user is parsed, and its MRS representation is converted into a discourse representation structure or **DRS** augmented with presuppositions (see Section 3.3). The DRS's presuppositions are then resolved using a model of the current discourse context (Section 3.4). If there are problems with presupposition resolution, various kinds of **error message** are passed to the dialogue manager; if not, a **resolved DRS** is passed forward. The dialogue manager then formulates an appropriate response (see Section 3.5). Before it generates the response, it updates its representation of the discourse context (the stack DRS, the context DRS and the saliency list), and possibly its own knowledge base (the global DRS and the knowledge graph)—see Section 3.6 for details of these representations.

The simplest kind of response the dialogue manager can give is a single utterance. A **sentence response representation** is produced, and passed directly to the sentence planning module. This representation is basically a partial MRS structure, missing the semantic material needed to generate referring expressions. The sentence planner calculates suitable referring expressions, using the global DRS, the context DRS and the saliency list (see Section 3.7 for details). Finally, the completed MRS is sent to the sentence generator, which produces a response sentence.

A more complex kind of response the dialogue manager can make is to produce an extended discourse rather than just a single sentence. In this case, it generates a **discourse response representation**. This representation is sent to the content selection module, which determines a set of facts to include in the discourse (see Section 3.8). These facts are then passed to the text planning module, which builds them into a **discourse structure tree**—a hierarchical representation of a piece of text with sentence representations as its leaves. Finally, sentence representations are passed in turn to the sentence planner, where suitable expressions for signalling discourse structure are added, and suitable referring expressions are computed, and the resulting MRSs are sent to the sentence generator.

3.2 Sentence parsing/generation

For sentence parsing and generation, we have used the LKB (Linguistic Knowledge Building) system (Copestake *et al*, 2000). This system is bidirectional; sentence interpretation and sentence generation use the same declarative grammar.

3.2.1 Grammar modules

LKB is designed to operate with a unification-based grammar formalism. We have used an HPSG style grammar (Pollard and Sag, 1994). As mentioned in Section 2.3, our grammars for English and Māori are combined into a single grammar. This is done by associating each lexical item and grammatical construction with a `language` feature, whose value is either `english` or `maori`. A number of general HPSG rules are language-independent (for instance, the Head Feature principle and the Head-Complement rule); the `language` feature is unspecified for such rules. We also leave the `language` feature unspecified for proper names. Agreement requirements on the `language` feature make it impossible to parse or generate sentences containing a mixture of words from different languages. One useful feature of a combined grammar of this sort is that our system can parse sentences in either language, without having first to identify which language they are in. See the companion paper for details of our syntactic treatment of a fragment of English and Māori.

3.2.2 MRS representations

The semantic formalism used by LKB is called **Minimal Recursion Semantics (MRS)** (Copestake *et al.*, 2001). MRS is a formalism designed to be compatible with feature-based grammar formalisms, and to provide a 'flat' (and therefore tractable) semantic representation for generation systems. For our purposes, its main useful feature is that it supports the kind of representations used in contemporary semantic theory, such as the tripartite quantifier-based represesentations of NPs mentioned in Section 2.4. It also allows us to represent quantifier scope ambiguity. For instance, it allows us to represent the two possible scopings of the sentence *A man loves every woman.* (At present, however, we don't have an algorithm for resolving scope ambiguities; we simply generate all interpretations, and pick the first one.)

3.3 MRS-to-DRS conversion

The scoped MRS representation delivered by the parser is then converted into a representation which makes the discourse characteristics of the sentence more explicit, namely a discourse representation structure or **sentence DRS** (Kamp and Reyle, 1993) with presuppositions represented as a special kind of sub-DRS (van der Sandt, 1992). MRS representations are not exactly DRSs, because they do not associate sets of referents with the scopal elements introduced by quantifiers, and they do not represent DRT's notion of the accessibility relations between sub-DRSs in a complex DRS. Moreover, there is no explicit distinction between the presupposed elements of an MRS and the asserted elements. However, all of these elements can be retrieved from an MRS.

3.4 Presupposition resolution

When a DRS-with-presuppositions has been created, its presuppositions (including ordinary anaphoric expressions) are resolved to create a **resolved sentence DRS**, using a **context DRS** and a **saliency list** of referring expressions, both of which have been built up since the start of the discourse. (The saliency list currently contains a list of referents in order of decreasing recency, tagged with number and gender.) The presupposition module first finds all possible ways of resolving the DRS's presuppositions in the discourse DRS. If there are none, its output is used to generate an appropriate follow-up utterance. If there is more than one, the saliency list is consulted. If there is a sufficient difference between the saliency of the top candidate and the next candidate, the presupposition can be resolved automatically; othewise its output is used to generate a different type of follow-up utterance (along the lines of *Which X?*).

3.5 Dialogue manager

The dialogue manager takes as input the result of presupposition resolution: either a resolved DRS or an error message. In the latter case, it manager formulates an appropriate follow-up question. In the former case, it first determines what dialogue act is being performed by the incoming sentence. If the dialogue act is inappropriate (for instance an answer when no question has been asked), a suitable regulatory response is formulated. If it is appropriate, a suitable response is formulated, according to a simple model of dialogue exchanges—for instance, questions are answered and assertions are acknowledged. For details about the operation of the dialogue manager, including more about the presupposition resolution module and the dialogue structures created by followup questions, see de Jager *et al.* (2002).

3.6 Context and knowledge updates

An important function of the dialogue manager is to update the system's representation of the discourse context after each dialogue move by the user. For assertive dialogue acts, this involves adding information to the context DRS, and updating the saliency list. It also may involve adding information to the global DRS.

We maintain a context DRS and a global DRS separately to allow the system to begin an interaction with the user with some knowledge up-front. Knowledge which the system has which was not gained during the present dialogue will appear in the global DRS but not in the context DRS. Such knowledge will have been gained in a previous dialogue, probably with a knowledge author rather than with a user. The user's questions are answered by consulting the global DRS rather than the local DRS, so that users can ask questions to which they do not know the answers.

3.7 Sentence planner

If the dialogue manager's response is a single sentence, the sentence planner is directly invoked. It is passed an MRS representation in which the semantic material needed to generate every referring expression has been removed and replaced with a pointer to a referent in the global DRS. The sentence planner then decides on a suitable referring expression to use for each DRS referent, using an algorithm similar to that given in Dale and Reiter (1992), and appropriate material is added to the MRS.

3.8 Content selection/text planning

The dialogue manager can also choose to respond with a multisentence text. The text will either be a paragraph describing one of the referents in the global DRS, or a text organised around a key fact mentioned in this DRS. In each case, the content selection module has to determine which information to include in the text, and the text planning module has to determine how to organise this information into a coherent discourse. Both these tasks are difficult to perform only using the information contained global DRS, since this information is simply an unordered collection of predicates about DRS referents.

To facilitate content selection and text planning, the system also maintains a resource called a **knowledge graph**, which contains all the information in the global DRS, stored in clause-sized units designed to support flexible sentence generation called **fact nodes**. The global DRS and the knowledge graph are created by the same authoring process. When the user enters a declarative sentence, information is added to the global DRS and to the context DRS as described in Section 3.6; in addition, a new fact node is added to the knowledge graph. The fact node is basically the MRS produced from the sentence with any context-specific material stripped out. In particular, material associated with each referring expression is removed and replaced with the appropriate global DRS referent. In other words, a fact-node is the kind of representation which can be passed to the sentence-planning module, which will construct suitable referring expressions for these DRS referents in any given discourse context.

As well as fact nodes, the knowledge graph contains two other kinds of nodes—**entity nodes** representing global DRS referents involved in fact nodes and **relation nodes** representing discourse relations between fact nodes. The content selection and text planning algorithms are both expressed as graph traversal algorithms on the knowledge graph, in the way described in O'Donnell *et al.* (2001).

4 A sample dialogue with Te Kaitito

An example of a dialogue with Te Kaitito is given in Figure 2. The user creates a small knowledge base by typing in declarative sentences (e.g. (a)) and then types in questions to query the knowledge base (e.g. (b)). Answers are generated in both English and Māori (e.g. (c)). The dialogue model currently requires explicit acknowledgement backchannels after assertions are understood (e.g. (d)). If the user enters a sentence with an ambiguous referring expression (e.g. (f)), the system generates a follow-up question (e.g. (g)). Note that the user could have switched to Māori at any point in the dialogue, either when information was being added or queried.

References

Barwise, J. and Cooper, R. (1981). Generalized quantifiers and natural language. *Linguistics and Philosophy*, **4**(2), 159–219.

Copestake, A. (2000). The (new) LKB system. CSLI, Stanford University.

Copestake, A., Lascarides, A., and Flickinger, D. (2001). An algebra for semantic construction in constraint-based grammars. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL 2001)*, Toulouse, France.

Creaney, N. (1996). An algorithm for generating quantifiers. In *Proceedings of 8th International Natural Language Generation Workshop*, Sussex, UK.

Cunningham, H., Maynard, D., Bontcheva, K., and Tablan, V. (2002). GATE: a framework and graphical development environment for robust NLP tools and applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02)*, pages ??–??, Philadelphia.

Dale, R. (1988). *Generating Referring Expressions in a Domain of Objects and Processes*. Ph.D. thesis, Centre for Cognitive Science, University of Edinburgh.

Dale, R. and Reiter, E. (1992). A fast algorithm for the generation of referring expressions. *Proceedings of COLING*.

Dale, R., Oberlander, J., Milosavljevic, M., and Knott, A. (1998). Integrating natural language generation and hypertext to produce dynamic documents. *Interacting with Computers*.

de Jager, S., Knott, A., and Bayard, I. (2002). A DRT-based framework for presuppositions in dialogue management. In *Proceedings of the 6th workshop on the semantics and pragmatics of dialogue (EDILOG 2002)*, Edinburgh.

Heim, I. (1982). *The semantics of definite and indefinite noun phrases*. Ph.D. thesis, University of Massachusetts. Distributed by Graduate Linguistic Student Association.

Speaker	**Utterance**	
User	a blue dog barked	(a)
System	Ok	
User	a red dog walked	
System	Ok	
User	which dog walked	(b)
System	the red dog it was the red dog nä te kurï whero	(c)
User	Ok	(d)
User	the blue dog chased a green cat	(e)
System	Ok	
User	a weasel barked	
System	Ok	
User	a big cat chased the weasel	
System	Ok	
User	the cat barked	
System	Ok	
User	the red dog chased the weasel	
System	Ok	

User	the blue dog walked	
System	Ok	
User	the dog ate the weasel	(f)
System	which dog ate the weasel	(g)
User	the red dog	
System	Ok	
System	Ok	
User	the blue dog chased a slow monkey	(h)
System	Ok	
User	a loud monkey ate the blue dog	
System	Ok	
User	which monkey did the blue dog chase	
System	the slow monkey it was the slow monkey nā te makimaki pōturi	
User	Ok	
User	what chased the monkey	(i)
System	the blue dog it was the blue dog nā te kurī kikorangi	

Figure 2: A dialogue with Te Kaitito

Horacek, H. (1996). A new algorithm for generating referential descriptions. In *Proceedings of ECAI*, pages 577–581.

Kamp, H. and Reyle, U. (1993). *From discourse to logic*. Kluwer Academic Publishers, Dordrecht.

Neumann, G. (1994). *A uniform computational model for natural language parsing and generation*. Ph.D. thesis, University of Saarbrücken.

O'Donnell, M., Mellish, C., Oberlander, J., and Knott, A. (2001). ILEX: an architecture for a dynamic hypertext generation system. *Natural Language Engineering*, **7**.

Paris, C., Vander Linden, K., and Lu, S. (2001). Automatic generation of on-line help: a system based on practical issues. In *Proceedings of the 2001 Australasian Natural Language Processing Workshop*, Macquarie University, Sydney.

Piwek, P., Evans, R., and Power, R. (1999). Dialogue as knowledge editing. Technical Report ITRI-99-11, Information Technology Research Institute, University of Brighton.

Pollard, C. and Sag, I. (1994). *Head-Driven Phrase Structure Grammar*. University of Chicago Press.

Power, R. (1999). Controlling logical scope in text generation. In *Proceedings of the European Workshop on Natural Language Generation*, Toulouse, France.

Power, R., Scott, D., and Evans, R. (1998). What you see is what you meant: direct knowledge editing with natural language feedback. In *Proceedings of the 13th European Conference on Artificial Intelligence (ECAI)*, Brighton, UK.

Reiter, E. and Dale, R. (2000). *Building natural language generation systems*. Cambridge University Press.

Shieber, S. (1988). A uniform architecture for parsing and generation. In *Proceedings of the 12th International Conference on Computational Linguistics (COLING)*, Budapest.

Thompson, H., Tobin, R., McKelvie, D., and Brew, C. (1997). LT XML. software API and toolkit for xml processing. `http://www.ltg.ed.ac.uk/software`.

Van der Sandt, R. (1992). Presupposition projection as anaphora resolution. *Journal of Semantics*, **9**, 333–377.

Van Noord, G. (1993). *Reversibility in Natural Language Processing*. Ph.D. thesis, University of Utrecht, The Netherlands.