CORECT: Combining CSCW with Natural Language Generation for Collaborative Requirements Capture

John Levine* and Chris Mellish†

Department of Artificial Intelligence,
University of Edinburgh,
80 South Bridge,
Edinburgh EH1 1HN, Scotland, UK.

Fax: +44 31 650 6516

Keywords: applications-oriented NLG, tailoring in NLG

Word count: 6,600

In the Collaborative Requirements Capture Tool (CORECT) project, we are building a computer-based requirements capture tool for custom-built electronic testing systems. The requirements capture process involves the participation of a wide range of different types of people – the customer, the salesperson, systems engineers, quality assurance, marketing, and so on. Our aim is to build a Computer-Supported Cooperative Working (CSCW) system which will allow these participants to define an Automatic Test System (ATS) collaboratively by adding data and making changes to an evolving design. The collected information about the design will form a large knowledge pool, all of which is pertinent to the design as a whole, but most of which is irrelevant to any particular person engaged in the design process. We will therefore be using natural language generation (NLG) technology to create documents from the central knowledge pool which are tailored to the particular information needs of the participants. These documents will give the users a snapshot of the developing design and will enable them to see how it can be improved and further developed. This paper gives an introduction to the problem we are tackling and how we are trying to solve it, and argues that combining CSCW for input with NLG for output in this way solves some of the problems which are encountered when trying to use either technology on its own.

*Email: J.Levine@ed.ac.uk
†Email: C.Mellish@ed.ac.uk

1 Introduction

In the field of custom-built electronic systems, the requirements definition process from the initial draft specification to the final agreed design is demanding, and requires input from a wide range of skilled personnel. However, due to the lack of a centralised store of knowledge about the developing design, it can also be slow and prone to expensive oversights. This research project, being carried out in collaboration with Racal Research Ltd., Racal Instruments Ltd., Intelligent Applications Ltd., and the University of Sussex, seeks to investigate the automation of requirements capture and the creation of a database of information for system specification and documentation. The system we are developing is a Collaborative Requirements Capture Tool (CORECT) for use by all the participants in the design process, including the customer, the salesperson and the systems engineer.

At the time that this paper is written, we are at the start of what is to be a three-year project, so much of what will be said here concerns our initial ideas about the problem and how we intend to solve it. We will also be presenting our thoughts on how generated documents can be tailored to the individual needs of the various users, and on how we think that Computer-Supported Cooperative Work (CSCW) and natural language generation (NLG) can be usefully combined. Our first prototype for CORECT will be based on the tool for authoring knowledge bases which was developed as part of the IDAS (Intelligent Documentation Advisory System) project (Reiter et al., 1992, 1993). The controlled acquisition of information by this authoring tool will help to ensure that the specification is consistent and (eventually) complete. The tool will also give designers rapid feedback and make requirements information immediately available, helping customers, designers, managers and salespeople to work together by helping them to communicate better.

The role of the University of Edinburgh in this project is the development of a natural language generation component which can automatically derive various kinds of specification documents from the common underlying database. The constraints of document generation will impact on the format and contents of the database as much as the functionality expected of the specifications (e.g. verification and validation). This is an important consideration, because it is not always possible to support NLG from an application program if the needs of NLG are not taken into account as the system itself is designed (Swartout et al., 1991). In CORECT, we will be using NLG technology to create the documents for the various participants in the design process, such as the customer, the salesperson and the design engineers. Since these users have radically different information needs, as well as different areas of expertise and vocabulary, we will using user modelling techniques to tailor the generated documents to the particular type of user they are intended for.

The use of natural language generation as part of our requirements capture tool also has the following potential advantages:

- NLG can produce documentation of the developing and completed design which is needed to meet existing legal and contractual requirements. Under the current paper-based practices, these documents have to be written by hand, although the content of these documents is such that at least large parts of them could be generated automatically from an underlying representation of the design.
- Generated documents make it easier for people outside the development team to examine and comment on the design and also make it easier for the system to be integrated with existing paper-based methods. This is an important point, since it is very unlikely and probably undesirable that an organisation involved in the design of custom-built equipment would be able to change its method of working overnight.
- The computer-based representation of the system which is the result of the requirements capture process can be used as a starting point for the creation of a knowledge base

for the user documentation of the system. That is, a suitably enriched version of a knowledge base created by CORECT could be used as a knowledge base for a system such as IDAS, which produces on-line documentation about the components of the ATS for the various types of users of this machine. In the IDAS project, we found that the cost of authoring the knowledge needed to provide tailored documentation outweighed the benefits of having a context-sensitive documentation system (Reiter and Mellish, 1993; Reiter et al., 1993). However, if most of the data needed for generation could be gained for free, then the cost of authoring would be much reduced and hence the costs-benefits tradeoff would become much more favourable.

- Automatic natural language generation techniques can produce documents adhering to strict content or style standards, such as Simplified English (AECMA, 1986). These standards are often imposed by customers or regulators, especially in the aerospace industry. Although the generation of texts in Simplified English will not be one of the major goals of the CORECT project, the philosophy of generating texts in plain language to make comprehension easy in complex environments is both interesting from a research perspective and likely to be of benefit in application programs.

The problem domain in which CORECT will operate is the collaborative design of an Automatic Test System (ATS). Such devices are designed and manufactured by Racal Instruments in direct response to customer requirements for automated electronic testing of complex equipment. The ATS mainly consists of modular industry-standard computer-controlled instrumentation but each system is different and often complex. In particular, a given system may require the design of a novel piece of equipment to be integrated with the standard modular components. Because a relatively small number of test systems are produced in any given configuration, it is important that the requirements capture process should be swift and effective. In addition, because of the custom-built nature of these products, the cost of the documentation for the machine is a large part of the overall cost, and hence if at least part of the documentation could be generated automatically from the completed requirements specification, this would reduce the overall cost of the ATS. If the results are successful in solving the problem in the ATS domain, we will attempt to generalise them by applying the system to other complex system domains within the Racal Electronics Group.

2 Combining CSCW for Input with NLG for Output

The term Computer-Supported Cooperative Work (CSCW) refers to computer systems which enable a group of individuals to collaborate on a piece of collective work, such as the writing of a paper with multiple authors. Many hypertext systems already support asynchronous working between different people; in the Xerox NoteCards system (Irish and Trigg, 1989), multiple authors may open and read the same node, but only one user has the ability to modify the node's content at one time. The Aquanet system (Marshall et al., 1991), under development at Xerox PARC, is a hypertext tool to support collaborative knowledge structuring. In CORECT, we will be developing this idea so that different users will have their own views of the common data, improving communication effectiveness, and building the information at a fact level rather than a document level, from which individual documents can be generated.

Techniques for ensuring that the right information gets delivered to the right people at the right time have been of interest to CSCW since the field's beginnings, with perhaps the best-known project being the MIT Information Lens (Malone et al., 1987). These ideas were further developed in subsequent projects, including Object Lens (Lai and Malone, 1988), the CMU Advisor system (Borensten and Thyberg, 1991) and the GM/EDS InVision system (Kass and Stadnyk, 1992). The last of these, which distributes technical documents (engineering change notices) and uses advanced user-modelling techniques as well as production rules to filter the documents, is probably closest to what we are doing in CORECT.

The above-mentioned systems all simply distributed complete messages. In CORECT, however, our intention is to go beyond this by extracting information relevant to a particular user from the common knowledge pool, and then presenting this to the user as a natural language document. Other NLG systems that extract and summarise information have been developed in other research, particularly by CoGenTex; their systems include, for example, FOG (Bourbeau et al., 1990), which produced weather reports; LFS (Iordanskaja et al., 1992), which summarised employment statistics; and Joyce (Rambow and Korelsky, 1992), which summarised software designs from a security perspective. The work on Joyce is particularly interesting because part of its justification was that natural language design summaries are useful to the designers themselves, as well as to people outside the design group. We expect that designers will find summaries even more useful in a multi-author design tool such as CORECT, since they will give them an overview of the progress of the design as a whole, and of what their colleagues have accomplished to date.

The proposed combination of CSCW for collecting and modifying the knowledge pool together with NLG for presenting users with selective views of the data is one which potentially solves problems which are encountered when trying to use either technology individually. Research in CSCW to date provides us with the means to collect data asynchronously from a diverse collection of users and hold that data in a format in which consistency checking (i.e. verification and validation) can be performed. However, for many applications of this technology, such as the collection of requirements information proposed in CORECT, the pool of knowledge soon grows in size such that it is not possible to see all of the information at once. In addition, if the data has been collected and entered by a heterogeneous user group with diverse interests and information needs, then the vast majority of the information in the database will be irrelevant to any particular user. Since the requirements capture process is iterative, in the sense that a user will use a summary of the current design in order to improve and augment it, there is a need for CSCW systems in areas such as ours to be able to present selected information from the data pool for individual users. This role can best be filled using NLG technology to generate documents which are tailored to the needs of the individual user.

The first and probably the most important requirement for natural language generation is that the initial data required for generation, i.e. the domain knowledge, should be available. It is certainly possible to say that we can use NLG technology to generate different documents and texts from the same underlying data, but if the underlying data is not there or is impoverished in some way, then no NLG can take place. In the IDAS project, our goal was the automatic generation of on-line documentation for Automatic Test Systems and other complex custom-built equipment. The knowledge base for the IDAS generator contained enough information about the equipment being documented to support different styles of documentation for the different user tasks and expertise levels. During this project, it was realised that authoring the knowledge base by hand for a complex piece of equipment such as an ATS would be a difficult task, and so a purpose-built graphical authoring tool was developed which would enable systems designers to enter this data more readily. However, by the end of the project, our conclusions were that the benefits gained from the provision of user-tailored documentation were not sufficiently large to outweigh the cost of authoring the large knowledge bases required (Reiter and Mellish, 1993; Reiter et al., 1993).

Given this need for the knowledge required for natural language generation to be collected more cheaply, it makes sense to see whether the data used in other processes, such as the data used during the design of the equipment, could be used for NLG. In CORECT, we are taking this one stage further, by making NLG an integral part of a tool whose primary function is to capture requirements data. Therefore, in this particular application, as far as NLG is concerned, the data comes with no additional cost attached. In addition, the knowledge base constructed during the design process makes a very good starting point for the construction of a knowledge base for a user-oriented system such as IDAS. Although it would be necessary to add information which is not necessary for the design but which is vital for use, maintenance and repair of the machine, the data collected during the requirements capture process would provide a very useful skeleton for the creation of knowledge base for on-line user documentation. Therefore, the use of CSCW for the effective collection of data in CORECT has the potential for solving the authoring problem in natural language generation, at least for applications such as this one.

Looking beyond CORECT, the most important requirement for solving the problem of authoring data is that the data should either be automatically collected, or that the collection of data is done in order to serve some other computational purpose. The problem with many automatically collected sources of data, such as stocks and shares prices or physical measurements taken in a scientific experiment, is that a lot of work must be done in order to extract information from this data and present it as a summary in plain language. In addition, the techniques which can be used in order to perform this task tend to be very domain specific, with each domain having (for example) its own particular way of referring to points of interest and trends in the data. Structured data which is collected in order to serve some other computational purpose, such as data used in CAD tools or data stored in a conventional database, looks more promising as a potential source of data for NLG, since it is typically in a format which is reasonably close to a surface form, while being sufficiently conceptual to support different linguistic realisations of the same concepts. As long as this condition is met, then NLG techniques can be used to extract and summarise information from complex knowledge and data bases which already exist without incurring the large cost of authoring the domain data required for NLG.

3 An Overview of CORECT

The basic architecture for the CORECT system is shown in Figure 1. Each of the different types of user interacts with a graphical user interface, which allows the users to add components from a component store to the developing design. Each individual item in the component store is a terminal node of an IS-A hierarchy, which allows for the use of inheritance when defining the properties of individual components. For example, many automatic test systems are now built from strictly modular industry-standard instruments (VXI cards) which are mounted into a general-purpose instrument rack (VXI chassis) which connects the instruments together electrically. Since many of the physical properties of the VXI cards must be the same in order to allow this, these properties can be held as part of a generic definition whose properties are inherited by the individual instruments used in the ATS. The structure of the ATS being designed consists of a collection of components which are connected together, where an individual component may be a collection of subcomponents, all of which have to be authored in order to make up a large sub-system of the ATS itself. In essence, the user can pick up components from the parts store and either add them to to a developing parts hierarchy or block diagram showing connections.

The actual data corresponding to the component store, parts hierarchy and connections is held within the system's database. This is held in a form which is sufficiently detailed for consistency and coherency checking to be performed using expert system rules. The use of a central database of information which is examined and added to by the other three modules of the system is important, since the data pool can be regarded as the core of this system. Using this data-central architecture allows us to develop the system in a strictly modular way with the minimum number of interface specifications. This means that the database manager can be regarded as the minimal system, with the other three modules being extensions to this system which increase its functionality. This also means that if further modules are proposed, these can be added in much the same manner.

Considered together, the graphical user interface and the database management system provide the users with a system which can hold all the requirements data for a given design as

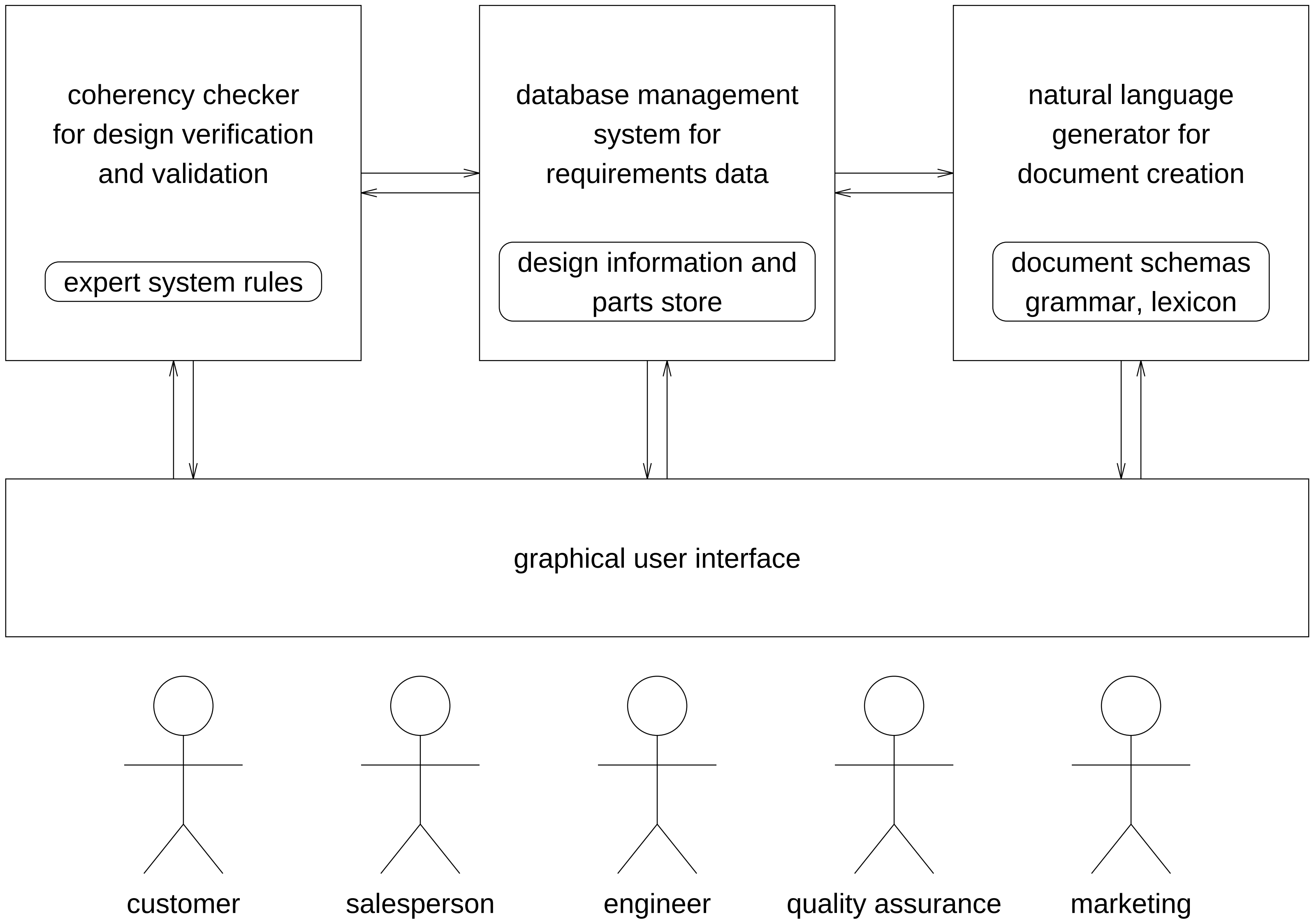

Figure 1: the architecture of the system

a centralised store. Current (i.e. paper-based) practices for requirements capture rely heavily on time-consuming effort to make sure that inconsistencies do not occur between the various documents about the system which are used to record information about the design. Therefore, the system consisting of just these two components has the potential to solve this problem, at least at a very basic level. However, the information held in the database may be incomplete or incoherent, and the system has no means for displaying the contents of the database to account for the diverse information needs and interests of the different users of the system.

The third component of the system is the natural language generator. This will be invoked by the user interface when the user requests that a particular document, such as a costing summary or a proposal, should be generated. The generator will select information from the database which is appropriate to this document, decide on how it should refer to the database concepts for this particular user, and then generate a final surface form for the document together with formatting directives (which could be in SGML or Latex, for example). The finished document will be returned to the user interface which will present it to the user on the screen or send it to be printed. The three phases of generation (content determination, sentence planning and linguistic realisation) will be broadly similar to those used in IDAS (Reiter et al., 1992) and in Joyce (Rambow and Korelsky, 1992). The output from the content determination phase consists of database propositions in which the individual terms and relations between them are written in the form in which they occur in the database itself. The sentence planner takes these raw propositions and decides on appropriate realisations for the terms and relations, so that the output from this phase could be (for example) an SPL expression (Kasper, 1989). Finally, the output from the sentence planning phase is turned into a natural language document by the linguistic realisation component.

The primary function of the NLG component in CORECT is to distribute information between the people who are engaged in the design process, allowing them to see different views of the data which are tailored to their particular needs. For example, a customer will be very interested in the overall cost of the machine, and in seeing that the functionality expected of the various components of the machine is met, and so a document prepared for this type of user should contain this sort of information with other more technical material being left out. One of our main aims in designing the CORECT NLG module is to investigate the issues involved in tailoring the content of what is said, and in finding a mechanism which is sufficiently powerful to allow a range of documents to be generated while also stressing that the methods used should be practical and implementable.

The final module of CORECT is the coherency checker, which will perform verification and validation checks on the design. Initially, this will be invoked manually by the user via the user interface, and it will then use expert system rules to see whether there are any gaps in the current design (i.e. components which still need to be added), and whether there are any inconsistencies in the current design, such as the wrong type of connecting cables being used. Considered as a whole, there are three aspects of the CORECT system which solve problems in collaborative requirements capture as it is currently practiced: (a) all the design data is kept in one place; (b) the system can provide different users with different views of this data using NLG; and (c) the system can provide verification and validation of the design, helping to minimise costly oversights.

4 Tailoring the User's View of the Database Using NLG

In generating documents which are intended to meet the requirements of a particular user, there are variations which can occur in all three phases of the generation process described above. In content determination, the NLG system can decide to include some facts and exclude others, which can include using different discourse strategies for different types of user, as in the TAILOR system (Paris, 1993). In sentence planning, the NLG system can choose different lexical units (Reiter, 1991) and construct different referring expressions (Reiter and Dale, 1992) depending on how much lexical competence the user has. In linguistic realisation, it is possible to phrase texts in a number of ways because different classes of users have different ways of speaking involving differences in the particular phrasal and grammatical patterns they employ (Bateman and Paris, 1989).

4.1 An Example from IDAS

The starting point for user tailoring in CORECT is what was achieved in the IDAS project. In IDAS, the tailoring of the output depends on two things: the user model and the user task. The user model is an indicator of the lexical and action competence of the user (i.e. whether or not this type of user will be expected to know technical terms and know how to perform complicated tasks). The user task is an indicator of the general type of activity in which this user is engaged (e.g. operations, routine-maintenance, repair-part). Each user then has a different view of the underlying database based on the values of these two parameters. This variation occurs in four ways. Firstly, we can offer different basic questions about a component of the machine, based on the user task. Secondly, we can vary the content of an answer to a particular question, also based on the user task. Thirdly, we can decide to describe an action to be performed as a large number of low-level steps or as a smaller number of higher-level steps, based on the user model. Finally, we can use different words to refer to objects and actions, also based on the user model. Together, these mean that different users may have quite different views of the underlying data.

To take an example, we will compare the view of the database offered to an ATS novice with an operations task with that offered to an ATS expert engaged in a repair-part task. The operations task means that this user is simply using the ATS to test a piece of equipment; typically this means that the user will connect the unit under test to the ATS, run a test program and collect the results of the test from the printer. This type of user is seldom called upon to do anything more than this, although the test program may instruct the operator to change the configuration of the machine in some minor way during its execution. In contrast, the repair-part task means that this user is repairing some of the components of the ATS itself, and so will need access to detailed information concerning the specifications and electrical connections of the machine.

To see how the content and wording of the documentation is tailored using the methods used in IDAS, we will look at one particular component of the machine, namely the computer, and see how the documentation offered varies for these two users. The first variation occurs in the range of questions which it is possible for these two users to ask: since the operator's task is essentially to locate components and use them, a limited range of questions are possible:

What is the computer?
Where is the computer?
How do I use the computer?

In contrast, the repair-part task is such that the repairer needs as much information about the ATS and its components as possible. This gives a much wider range of possible questions:

What is the computer?
Where is the computer?
What are the parts of the computer?
What are the specifications of the computer?
What is the purpose of the computer?
What is the computer connected to?
How do I repair the computer?

The second variation occurs in what content is used to answer a particular question. This also varies with the user task. In answering the 'What is the computer?' question for the operator, the salient properties of the computer are the colour, the manufacturer and the model number; this information is included in the answer so that the operator can identify the computer if the test program asks the user to perform some operation on the computer:

> What is the computer?
> It is a black HP/Apollo DN4500 computer.

For the repairer, we leave out the colour, since it is irrelevant for this task, but include the detailed model number and the parts number:

> What is the computer?
> It is an HP/Apollo DN4500-SX computer with part number OPT-AP8.

The third variation is concerned with explaining action sequences at an appropriate level and this depends on the user model. A novice user will need to have actions described in as much detail as possible, whereas a skilled user will only need a summary of the high-level actions to be performed. For the skilled user, the answer to the question 'How do I use the computer?' could simply be the following:

> How do I use the computer?
> Put a floppy disk into the computer's disk drive. Load the program. Run it.

However, for the novice, each of these actions would be expanded out into a number of lower-level steps:

> How do I use the computer?
> Open the latch of computer's disk drive. Insert a floppy disk into the disk drive. Close the latch. Type 'engage disk' on the computer's keyboard. Type 'load program' on the keyboard. Type 'engage program' on the keyboard. Collect the test results from the printer.

Here, the first action in the short description has been expanded into three lower-level actions, with the other two actions being expanded into two lower-level actions, making this text easier to follow for a novice user.

The final variation occurs in the words which are used to refer to knowledge base entities, and this depends on the user model. In answering the 'Where is the computer?' question, we may wish to say that it is above the mains control unit. For a skilled user this is fine, but a novice is unlikely to know what a mains control unit looks like. Therefore, since the mains control unit is a type of power supply and since the novice user knows this term we can phrase the answer to the question as follows:

> Where is the computer?
> It is above the silver power supply.

The additional information that the power supply is silver is added in order to distinguish it from the other power supplies present in the ATS.

4.2 Improving User Tailoring in CORECT

It can be seen from the examples above how IDAS tailors both the content and the words used in a particular part of its documentation. However, it should be noted that the taxonomy of user models and user tasks used here is based largely on intuition, and the variation in the texts shown above is also based on the same intuitions. Although we have a good idea that operators and repair personnel do exist, we have never verified that the information shown above is what these types of users actually need. At best, we can say that we have demonstrated that a large degree of tailoring is possible.

In CORECT, we intend to improve upon this by getting a firmer idea of what types of users there will be and what sort of views of the database these users will want. For the former, workplace studies are currently being carried out, one of whose objectives is to identify the different types of people who are involved in requirements capture. For the latter, the workplace studies will also reveal formal and informal documents that we can seek to emulate in terms of both vocabulary and content. We also intend to carry out extensive user trials of our system using real users to see whether our generated documents contain the sort of information which individual users actually need. These user trials will probably be qualitative in nature, unlike the quantitative user trials used to evaluate IDAS (Reiter et al., 1993), but they will still provide us with enough information to refine the user tailoring employed in the system to the point where we will be able to say that our documents genuinely address the individual needs of real users. Since, to the best of our knowledge, no research has ever attempted to confirm that current techniques for user tailoring, such as those used in IDAS, are actually of benefit to users, we imagine that this will be a worthwhile enterprise.

5 Advancing the State of the Art in NLG

In this project, our goal is to advance the state-of-the-art in applications-oriented natural language generation, and to demonstrate that such technology can be of great value when combined with CSCW. We are particularly interested in finding suitable application areas where NLG technology can be of real use, and in developing techniques which maximise the benefits of NLG while minimising the costs involved (Reiter and Mellish, 1993). In our opinion, one of the main potential application areas for NLG is in the selective extraction of information from large or complex databases and its presentation as clear natural language text. This functionality is especially important in CSCW systems that require participants with widely different backgrounds and interests to cooperate in general and share data in particular.

Within the area of natural language generation itself, our main goal is to continue our search for simple and implementable techniques that can be used to produce large amounts of text in a real application (Levine et al., 1991; Reiter et al., 1992; Reiter and Dale, 1992; Reiter and Mellish, 1993). CORECT should be an excellent vehicle for investigating these concerns in constructing whole documents rather than the small text chunks used in IDAS. We will be examining in detail the problems of content determination and document structuring which were not addressed in IDAS, and seeking to find practical, applications-oriented solutions to these problems. This may produce texts with inferior language to human-generated ones but the texts will at least be complete and consistent with the knowledge base. It remains to be seen to what extent the generation of correct first drafts is useful if they are inferior in this way, but the success of post-edited machine translation gives us hope in this area. We will also be trying to extend our work on user-modelling and tailoring of texts to allow for the wide variety of documents which CORECT will need to produce.

Finally, since we believe that evaluation of this technology is crucial for its further development, as well as for the basic scientific verification of the claims made in favour of these techniques, we will be placing strong emphasis on the evaluation of our generation system. In IDAS, this evaluation was done via a system of quantitative user trials, in which subjects were asked to navigate their way through the documentation in order to find the information they needed to complete a printed test paper (Reiter et al., 1993). By observing the way in which the users interacted with the system, we were able to conclude that the claims made in favour of a combination of NLG and hypertext (Levine et al., 1991) were essentially accurate. In CORECT, our aim is to extend and augment the techniques for evaluation of NLG systems which were developed for evaluation in the IDAS project.

6 Concluding Remarks

Under current practices, requirements definition for complex systems is extremely demanding and can lead to expensive oversights if the skills and disciplines required are not carefully and rigorously applied. Current methods rely heavily on skilled personnel and time-consuming effort to ensure that the understanding is universal and all aspects of the design and requirements have been addressed. What is needed is a centralised store of information about the design, in a format which will support verification and validation of the specifications. Since requirements capture is a collaborative exercise involving a diverse group of people, this implies that a computer-based requirements capture tool must support CSCW. While CSCW is sufficient for the collection of the data, it does not provide a solution to the problem that the pool of knowledge about the design will be large, and most of it will not be relevant to any one particular user. Hence, we will be using NLG technology to provide users with radically different views onto the same knowledge pool via documents which address their information needs. We propose that combining CSCW for input with NLG for output in this way will be a useful combination of technologies and will solve problems which are encountered when trying to use either technology on its own.

Acknowledgements

The CORECT project is partially funded by UK SERC grant GR/J/53454 and UK DTI grant IED4/1/7025, and we are grateful to SERC and the DTI for their support of this work. We would also like to thank the CORECT collaborators – Racal Instruments Ltd., Racal Research Ltd., Intelligent Applications Ltd. and the University of Sussex – for the help that they have given us in performing this research. We would also like to express our thanks to Dr. Ehud Reiter for providing many of the original ideas from which this research has developed.

References

AECMA (1986). A Guide for the Preparation of Aircraft Maintenance Documentation in the International Aerospace Maintenance Language. Available from BDC Publishing Services, Slack Lane, Derby, UK.

Bateman, J. and Paris, C. (1989). Phrasing a Text in Terms the User Can Understand. *Proceedings of the 11th International Conference on Artificial Intelligence (IJCAI–89)*, Volume 2, 1511–1517.

Borensten, N. and Thyberg, C. (1991). Power, Ease of Use, and Cooperative Work in a Practical Multimedia Message System. *International Journal of Man-Machine Studies*, 34, 229–260.

Bourbeau, L., Carcagno, D., Goldberg, E., Kittridge, R. and Polguere, A. (1990). Bilingual Generation of Weather Forecasts in an Operations Environment. *Proceedings of the 13th International Conference on Computational Linguistics (COLING–90)*, Volume 1, 90–92.

Iordanskaja, L., Kim, M., Kittridge, R., Lavoie, B. and Polguere, A. (1992). Generation of Extended Bilingual Statistical Reports. *Proceedings of the 14th International Conference on Computational Linguistics (COLING–92)*, Volume 3, 1019–1023.

Irish, P. and Trigg, R. (1989). Supporting Collaboration in Hypermedia: Issues and Experiences. *Journal of the American Society for Information Science*, 40(3), 192–199.

Kass, R. and Stadnyk, I. (1992). Using User Models to Improve Organisational Communication. *Proceedings of the Third International Workshop on User Modelling*, 135–137.

Kasper, R. (1989). A Flexible Interface for Linking Applications to Penman's Sentence Generator. *Proceedings of the 1989 DARPA Speech and Natural Language Workshop*, Philadelphia, 153–158.

Lai, K. and Malone, T. (1988). Object Lens: A 'Spreadsheet' for Cooperative Work. *Proceedings of the Conference on Computer-Supported Cooperative Work (CSCW '88)*, Portland, Oregon, 115–124.

Levine, J., Cawsey, A., Mellish, C., Poynter, L., Reiter, E., Tyson, P. and Walker, J. (1991). IDAS: Combining Hypertext and Natural Language Generation. *Proceedings of the Third European Workshop on Natural Language Generation*, Judenstein, Austria, 55–62.

Malone, T., Grant, K., Turbak, F., Broust, S. and Cohen, M. (1987). Intelligent Information-Sharing Systems. *Communications of the ACM*, 30, 390–402.

Marshall, C., Halasz, F., Rogers, R., Janssen, W. (1991). Aquanet: a Hypertext Tool to Hold Your Knowledge in Place. *Proceedings of the 3rd ACM Conference on Hypertext*, San Antonio, Texas, 261–275.

Paris, C. (1993). *User Modelling in Text Generation.* Pinter Publishers, London and New York.

Rambow, O. and Korelsky, T. (1992). Applied Text Generation. *Proceedings of the Third Conference on Applied Natural Language Processing*, Trento, Italy, 40–47.

Reiter, E. (1991). A New Model of Lexical Choice for Nouns. *Computational Intelligence*, 7(4), 240-251.

Reiter, E. and Dale, R. (1992). A Fast Algorithm for the Generation of Referring Expressions. *Proceedings of the Fourteenth International Conference on Computational Linguistics (COLING–92)*, Volume 1, 232–238.

Reiter, E., Mellish, C. and Levine, J. (1992). Automatic Generation of On-Line Documentation in the IDAS Project. *Proceedings of the Third Conference on Applied Natural Language Processing*, Association for Computational Linguistics, 64–71.

Reiter, E. and Mellish, C. (1993). Optimizing the Costs and Benefits of Natural Language Generation. *Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI–93)*, Chambery, France.

Reiter, E., Mellish, C. and Levine, J. (1993). Automatic Generation of Technical Documentation. Submitted to *Applied Artificial Intelligence.*

Swartout, W., Paris, C. and Moore, J. (1991). Design for Explainable Expert Systems. *IEEE Expert*, June 1991, 58–64.