Lessons from Deploying NLG Technology for Marine Weather Forecast Text Generation

Somayajulu G. Sripada[1] and **Ehud Reiter** [1] and **Ian Davy**[2] and **Kristian Nilssen**[2]

Abstract. **SUMTIME-MOUSAM** is a Natural Language Generation (NLG) system that produces textual weather forecasts for offshore oilrigs from Numerical Weather Prediction (NWP) data. It has been used for the past year by Weathernews (UK) Ltd for producing 150 draft forecasts per day, which are then post-edited by forecasters before being released to end-users. In this paper, we describe how the system works, how it is used at Weathernews and finally some lessons we learnt from building, installing and maintaining **SUMTIME-MOUSAM**. One important lesson has been that using NLG technology improves maintainability although the biggest maintenance work actually involved changing data formats at the I/O interfaces. We also found our system being used by forecasters in unexpected ways for understanding and editing data. We conclude that the success of a technology owes as much to its functional superiority as to its suitability to the various stakeholders such as developers and users.

1 INTRODUCTION

Modern weather service companies operate in a competitive market where quality of their forecasts must show continuous improvement. Forecasters in these organizations predict weather under the guidance of weather data generated by Numerical Weather Prediction (NWP) models. In order to produce a weather forecast for a specific end user, they carry out two tasks:

1. To compile weather prediction information fulfilling the needs of the end user. This task requires them to use NWP model along with other sources of weather data such as satellite pictures and their own forecasting experience.
2. To present the weather prediction information to the end user in a suitable medium such as graphics or text.

The quality of the weather prediction is largely determined by the first task. Particularly, the knowledge that human forecasters bring to weather forecasting is very crucial for higher quality forecasts. From the quality perspective, forecasters are expected to spend more time on the first task in comparison to the second task. This is possible if the second task of presenting weather information to the end user is automated.

In collaboration with Weathernews (UK) Ltd, as part of the **SUMTIME** project, we have studied the task of presenting weather information textually to oil company staff supporting offshore oilrig operations in the North Sea. We have used a variety of knowledge acquisition (KA) techniques developed in the expert system community to understand how humans perform weather forecasting [1].

[1] Department of Computing Science, University of Aberdeen, Aberdeen, Scotland, U.K., email: {ssripada,ereiter}@csd.abdn.ac.uk.

[2] Weathernews (UK) Ltd, Aberdeen, Scotland, U.K., email: {iand,kristiann}@wni.com.

From the KA we have identified a number of requirements that the text generation solution should fulfill. A few of these are described below:

1. **Consistency of Language Use**: Individual forecaster variation is one of the initial observations we have made while studying a corpus of human written forecasts [2]. For example, forecasters differed significantly in their usage of time phrases such as 'in the evening' to mean either 1800 hours or 2100 hours. This could cause confusion to the end user about when exactly a predicted change occurred. Using language consistently is an important requirement.
2. **Sensitivity to End-User**: Another key observation was that content of a forecast should depend upon the end user. Different oilrigs need different details of information in a weather forecast. An offshore oilrig in the North Sea might require different details from one in the Persian-gulf because of the differences in their structural designs.
3. **Forecaster Control**: It is important that the forecasters who use our system should be able to control its output without writing new code. The need for this might arise due to changes in end user requirements. Because of their long working experience, forecasters understand the end user requirements and also understand how they effect the generation of forecast texts.
4. **Data Analysis**: Weather data used by forecasters consists of time series of various parameters such as wind speed, and wind direction. Forecasters analyse the time series data to select important data points to include in the weather report. This data analysis needs to be integrated with text generation.

Using the knowledge gathered from our KA studies, we built **SUMTIME-MOUSAM** to generate textual marine weather forecasts for offshore oilrig staff [3]. The system has been in use at Weathernews for the past year producing draft forecasts, which are then post-edited by their forecasters before being sent to the end users. In section 2 we briefly describe **SUMTIME-MOUSAM** and explain how it is used at Weathernews. In section 3 we discuss the lessons we learnt from our experience of building, installing and maintaining **SUMTIME-MOUSAM**.

2 SUMTIME-MOUSAM

SUMTIME-MOUSAM follows the simple pipeline architecture for text generation [4] as shown in Figure 1. Input to **SUMTIME-MOUSAM** is obtained by sampling forecaster edited data from the NWP model prediction at the required grid point.

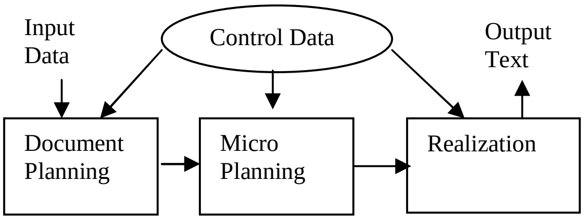

Figure 1. Architecture of **SUMTIME-MOUSAM**

Table 1 shows a portion of input to our system for 12 June 2002. Full data set includes approximately forty basic weather parameters predicted for 72 hours (3 days) from the issue time of the forecast. Table 2 shows the first day forecast text generated by our system. Output forecast text is organized into various fields such as Wind, Wave, Weather etc. Each of the fields describes a few basic weather parameters. For example, the wind field of the forecast has been generated using the data shown in Table 1. Next, we briefly describe the major modules.

Table 1. Weather Data produced by an NWP model for 12-Jun 2002

Time	Wind Dir	Wind Spd 10m	Wind Spd 50m	Gust 10m	Gust 50m
03:00	SW	9.0	11.0	11.0	14.0
06:00	W	10.0	12.0	12.0	16.0
09:00	W	11.0	14.0	14.0	17.0
12:00	WSW	10.0	12.0	12.0	16.0
15:00	SW	7.0	9.0	9.0	11.0
18:00	SSW	8.0	10.0	10.0	12.0
21:00	S	9.0	11.0	11.0	14.0
00:00	S	12.0	15.0	15.0	19.0

Document Planning: This stage is responsible for determining content and organizing (structuring) it coherently. We use Weathernews' recommended structure. Content determination involves selecting 'important' or 'significant' data points from the underlying weather data to be included in the forecast text using the bottom-up segmentation algorithm [5][6].

Micro-planning: This stage is responsible for lexical selection, aggregation and ellipsis. Here we have used the rules we have collected from our corpus analysis and other KA tasks.

Realization: Finally this stage is responsible for producing a grammatically acceptable output. We have developed a simple realizer that is tuned to the sublanguage of weather forecasts.

Table 2. Forecast Text Produced by **SUMTIME-MOUSAM** for the AM of 12-Jun 2002. The Wind part of the forecast has been generated from the data, shown in Table 1

Section 2. FORECAST 6 - 24 GMT, Wed 12-Jun 2002	
Field	**Text**
WIND(KTS) 10M	W 8-13 backing SW by mid afternoon and S 10-15 by midnight.
WIND(KTS) 50M	W 10-15 backing SW by mid afternoon and S 13-18 by midnight.
WAVES(M) SIG HT	0.5-1.0 mainly SW swell.
WAVES(M) MAX HT	1.0-1.5 mainly SW swell falling 1.0 or less mainly SSW swell by afternoon, then rising 1.0-1.5 by midnight.
WAVE PERIOD (SEC)	Wind wave 2-4 mainly 6 second SW swell.
WINDWAVE PERIOD (SEC)	2-4.
SWELL PERIOD (SEC)	5-7.
WEATHER	Mainly cloudy with light rain showers becoming overcast around midnight.
VISIBILITY(NM)	Greater than 10.
AIR TEMP(C)	8-10 rising 9-11 around midnight.
CLOUD (OKTAS/FT)	4-6 ST/SC 400-600 lifting 6-8 ST/SC 700-900 around midnight.

Control Data: This is an external data source that partially controls the text generation process in **SUMTIME-MOUSAM**. Forecasters can edit this data externally to tailor the output text. For example, error function data for controlling the segmentation process is specified here.

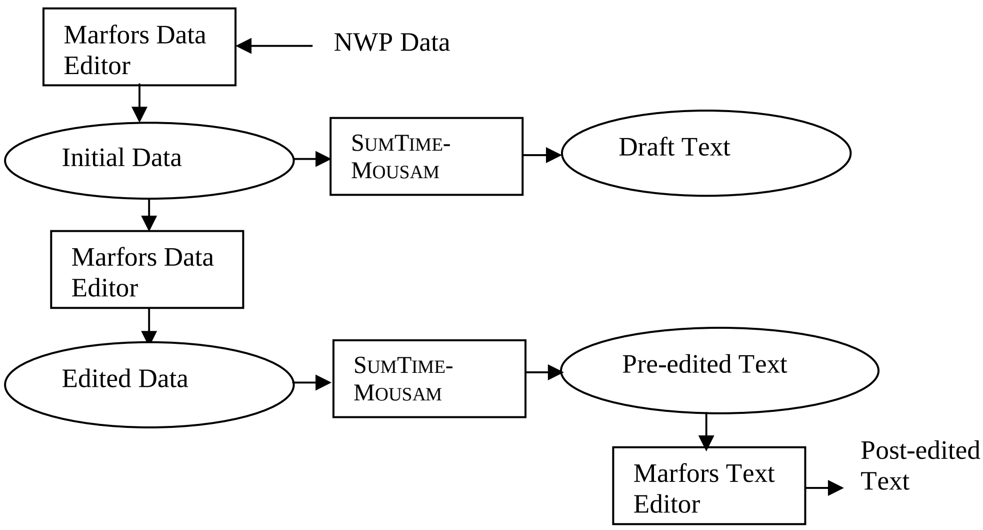

Figure 2. Schematic showing **SUMTIME-MOUSAM** used at Weathernews

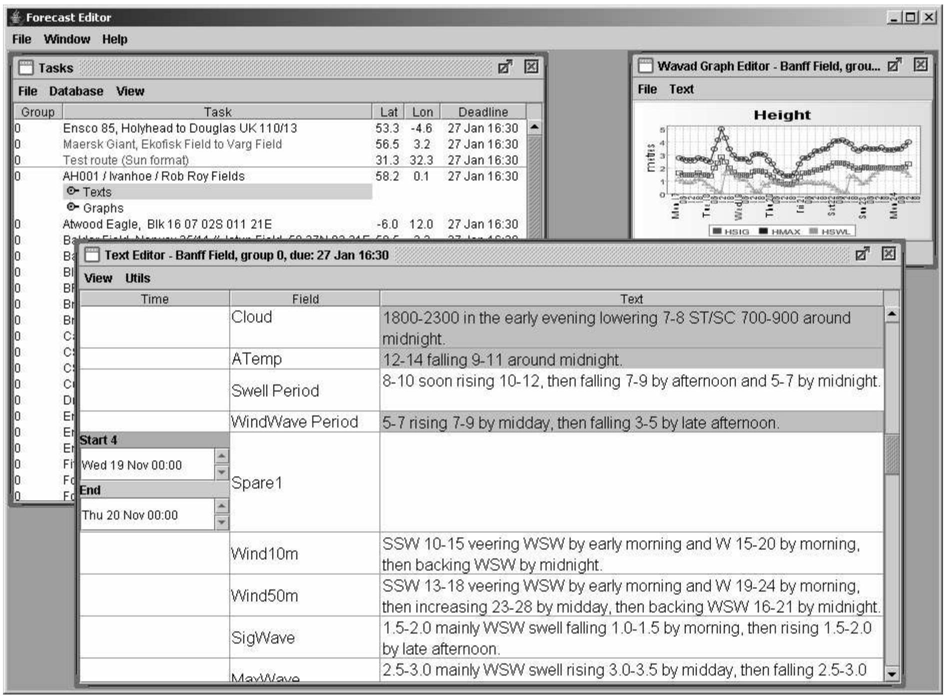

Figure 3. A screen shot showing the text editor used by forecasters for editing the text generated by **SUMTIME-MOUSAM**.

Weathrenews manages their marine forecast production with the help of an internally developed tool called Marfors. Marfors calls **SUMTIME-MOUSAM** to produce textual forecast from weather data (NWP model output or forecaster edited data). Figure 2 shows the use of **SUMTIME-MOUSAM** at Weathernews. To start with forecasters initially load the NWP data corresponding to a client's request into Marfors. They then use their meteorological knowledge to edit the NWP data and call **SUMTIME-MOUSAM** to generate an initial draft of the textual forecast for the required location. Forecasters use this initial draft largely to understand the underlying data set. This has been an unexpected use of our system; our system output was not intended for forecasters. Yet, we find the forecasters using our output to understand data and to use that understanding for editing data. The cycle of edit-data and generate-text is carried out for a number of iterations until the forecasters are satisfied with the weather data, which is shown as '*Edited data*' in Figure 2 **SUMTIME-MOUSAM** is once again invoked to generate a final draft textual forecast, which is shown as '*Pre-edited text*' in Figure 2. Forecasters use Marfors to post edit the draft textual forecast to prepare the final forecast, which is shown as '*Post-edited text*' in Figure 2. Figure 3 shows the screen shot of Marfors editors used by forecasters at Weathernews to edit data and text.

In the past there have been efforts to use NLG technology for weather forecasting. For example ICWF [7], FOG [8] and MultiMeteo [9]. For a more exhaustive list of weather forecast text generators please refer to the NLG system list published by John Bateman and Michael Zock [10]. Of particular significance is the system FOG (Forecast Generator) used by the Canadian Weather Agency. FOG produced public and marine weather forecasts in two languages, English and French. The main focus of the system has been on using Meaning Text Theory to generate bilingual output. Based on the descriptions of the system, it is not clear how it performs content selection. In comparison to FOG where the focus is more on the linguistic theory for bilingual output, **SUMTIME-MOUSAM** focused more on analysis of weather data to determine 'important' information to be included in the weather report. MULTIMETEO is another weather forecast text generation system that follows an interactive approach to multi-lingual text generation. The system is equipped with a knowledge administration facility that allows forecasters to edit the output text generated in one language and uses that knowledge to produce forecasts in several other languages. Perhaps because of its commercial value, it is hard to find published material describing technical details of MULTIMETEO. Based on the available details it appears that our control data is similar in concept to their knowledge administration.

3 LESSONS FROM SUMTIME-MOUSAM

In the past one year **SUMTIME-MOUSAM** has been used by forecasters at Weathernews to produce 150 forecasts per day. We have carried out a post-edit evaluation of our system where we have counted the number of edits (add, delete, replace operations) human forecasters performed while editing 'pre-edited text' to produce the final 'post-edited text' as shown in Figure 2. For example consider the following texts:

A. Pre-edit Text: SW 20-25 backing SSW 28-33 by midday, then gradually increasing 34-39 by midnight.
B. Post-edit Text: SW 22-27 gradually increasing SSW 34-39.

We first divide A and B into constituent phrases such as '*SW 20-25*' and '*backing SSW 28-33 by midday*'. Phrases from A are then aligned to phrases from B. Once the phrases are aligned, we then count all the edits performed on each aligned phrase pair such as '*then gradually increasing 34-39 by midnight*' aligned to '*gradually increasing SSW34-39*'. For this example pair, we count '*then*' deletion, '*SSW*' addition and '*midnight*' deletion. More details of our evaluation work have been described in [11]. The results of our evaluation have been shown in the following table by classifying the mismatches:

S. No.	Mismatch Type	%
1.	Ellipses (word additions and deletions)	65
2.	Data Related Replacements (range and direction replacements)	20
3.	Lexical Replacements	15

According to our evaluation our rules for performing ellipsis need to be refined. Work is currently underway to learn better ellipsis rules. In this section we present the lessons we learnt at different phases of **SUMTIME-MOUSAM**'s lifecycle.

3.1 Lessons from the development phase

The very first version of **SUMTIME-MOUSAM** was developed based on a method suggested by one of the experts from Weathernews. This method used what can be called a template-based approach to text generation [12]. Essentially a template-based approach is based on manipulation of character strings to produce text output. There is no explicit linguistic knowledge in these systems and also there is no modularization of text generation tasks. Code belonging to a deeper level task such as selecting content also deals with a surface level task such as punctuation. Each separate output text is produced as a special case. Although this approach was simple to implement for the initial version, it was not easy to extend it to generate the full range of output texts. Particularly writing code for arranging words in the grammatical order and adding punctuation marks was a nightmare. Subsequent versions of **SUMTIME-MOUSAM** followed more modularised approach as described in the previous section. This allowed us to focus on each of the tasks of text generation independently. Also it facilitated our knowledge acquisition activities.

3.2 Lessons from the installation phase

The main work at this phase was to tune the I/O interface to Weathernews database. We should have anticipated this mismatch of I/O interfaces and taken corrective measures during the development phase. One solution we have followed finally was to de-couple I/O from the rest of the processing so that any future changes will affect only the I/O interface classes as long as the new classes pass the required data to the rest of the system. Although this is a routine software engineering issue, it is quite important for the success of our system.

3.3 Lessons from the maintenance phase

During the past one year of its operation at Weathernews, forecasters raised a number of concerns about our system output and we have carried out multiple maintenance activities on the system to fulfill these change requests. Through this work, we have made interesting observations about maintainability of NLG technology. Maintenance of a text generator, like for any other software, is an important phase of its life cycle. The FOG system discussed in section 2 has to deal with a number of sub-language issues during its maintenance phase [13]. The developers of the MULTIMETEO have designed knowledge administration station to allow forecasters to carry out routine maintenance operations on their system [14].

An important feedback from the forecasters using our system is that we should focus our future work more on simple fields we already do well rather than working with complex fields that we do only moderately well. In the context of building successful AI applications, Rob Milne [15] made a similar observation about focusing on 'narrow, vertical application areas' and recommended delivering 'complete solutions to users'.

1. **Database Issues**: As discussed earlier in sub-section 3.2, many changes were made to the system to alter its database. In fact, these changes took more time than all the other changes carried out on our system. Because text generation systems often function as embedded components in a larger system, interfaces to other components need to be carefully planned.
2. **User Interface Issues**: One of the main features of **SUMTIME-MOUSAM** has been that forecasters (users of the system) can control the output text by changing data in an external file (shown as Control data in Figure 1). After its installation, forecasters have experimented with control data and generated three different versions of control data files. They are called 'fine', 'default' and 'coarse'. As their names indicate these three files produce texts with different levels of detail, 'fine' produces a very detailed forecast while 'default' produces a forecast with lesser details and so on. Out of these files only 'default' gets used all the time understandably because it provides a balance between the other two. However, what is not clear is why forecasters do not use the control data more dynamically controlling individual forecasts. It was hard to get feedback on this from forecasters. Although control data was a feature requested by the forecasters, in practice they are not using it. Our current graphical user interface does not allow forecasters to edit control data while editing forecast text; the control data needs to be edited offline. Better user interface will perhaps make forecasters use this feature more often.
3. **Additional Fields in the Weather Report**: As discussed in section 2, weather forecasts are organised into a number of fields each describing data related to a subsystem of weather. We have been asked to add two new fields viz., 'Swell Period' and 'Wind Wave Period' (shown in Table 2). These two statements partially describe the information contained in the field 'Wave period' (also shown in Table 2). This change is a typical maintenance request asking for extending the system to produce additional text after its design and development. In our case, this extension did not force many changes to the system. This is because we already have all the stages of processing implemented in independent modules. We could simply call these modules to work on the required data without writing any new code that performs generation tasks such as content selection and lexical choice. On the other hand we like to contrast this with the amount of new code that would have been needed in template-based systems that we have talked about earlier in sub-section 3.1. For every new output, these systems need additional code written entirely from scratch performing tasks such as content selection and lexical choice. NLG techniques enjoy this advantage of building on the infrastructure developed initially for subsequent extensions.
4. **Improved NLG**: We have made many changes to our system based on the post-edits forecasters performed on our system output.
    a. Our study of post-edits revealed that the most common editing operation performed by the forecasters is to delete a word or phrase generated by our system. In other words, our rules for ellipsis require modification. In our system all the ellipsis rules are processed in a module

called micro-planner (see figure 1) which contains rules such as:

- Suppress direction phrase if same as the previous direction phrase
- Suppress speed phrase if same as previous speed phrase
- Suppress the entire wind phrase if both speed phrase and direction phrase are suppressed

Making changes to our system's ellipsis behaviour involves making changes to one of these rules located in a single module. On the other hand, template-based systems will require changes to be made at multiple locations leading to all the different outputs. From the maintenance perspective once again we find NLG technology more beneficial as opposed to the template technology.

b. One outstanding change request from forecasters requires our system to restrict the number of words contained in a statement. Human readers prefer a concise message carrying important weather information rather than a verbose statement. Such size constraints on text generation output have also been reported in [16]. Size constraints are hard to be fulfilled in the simple pipeline architecture we follow. Reiter [16] suggests that allowing multiple solutions to be passed down the pipeline or adding an additional revision module at the end are two alternative solutions to this problem. We plan to follow the multiple solution approach in our system.

   Multiple solutions can be useful in another context as well. When we generate text from weather data, there are multiple ways in which we can describe data. The current version of our system produces only one of these descriptions as its output. While post-editing our system output as described in section 2, we have noted some forecasters making lot more edits than others because of differences in their individual preferences. If our system generated multiple descriptions then forecasters could choose the required descriptions.

c. Another outstanding change request from Weathernews is about extending the current system to produce multi-lingual output, particularly in Norwegian. Because of our approach to text generation, our processing is language independent and therefore is capable of generating multi-lingual output.

d. Certain input data sets contain patterns of special interest and their description should be different from normal descriptions. For example, if wind speed shows a pattern of rise and fall many times in a forecast period, it should be described with the phrase 'variable wind' rather than a sequence of phrases describing wind 'rising' and 'falling'. We need additional data analysis techniques for detecting these patterns.

e. A related issue comes from an observation we made while studying humans writing forecast texts from weather data. Human forecasters analyze input weather data with a view to building what we call 'overview' of the weather. Forecasters use this overview to select content for individual fields. Because all the fields derive content consistent with the overview, the forecast as a whole conveys a consistent view of the weather. Overview is an elusive concept and we are exploring ways to incorporate it into our processing.

4 CONCLUSION

Text generators like any other software operate in an ever-changing environment and therefore must be designed for maintainability. From our experience of using **SUMTIME-MOUSAM** for weather forecast generation we conclude that NLG technology (a) allows forecasters to control the text without writing code and (b) offers robust design that allows easier maintenance. We also believe that the final success of the system depends upon its suitability to various stakeholders such as developers and users.

ACKNOWLEDGEMENTS

Many thanks to all the staff at Weathernews (UK) Ltd. for their active co-operation throughout this work; this work would not be possible without them! This project is supported by the UK Engineering and Physical Sciences Research Council (EPSRC), under the grant GR/M76881.

REFERENCES

[1] E. Reiter, S. Sripada, and R. Robertson, 'Acquiring Correct Knowledge for Natural Language Generation.' *Journal of Artificial Intelligence Research*, **18**: 491-516, (2003).

[2] E. Reiter and S. Sripada, 'Human Variation and Lexical Choice.' *Computational Linguistics* **28**:545-553, (2002).

[3] S. Sripada, E. Reiter, and I. Davy, 'SumTime-Mousam: Configurable Marine Weather Forecast Generator', *Expert Update*, **6(3)**, pp 4-10, (2003)

[4] E. Reiter and R. Dale, *Building Natural Language Generation Systems,* Cambridge University Press, Cambridge, 2000.

[5] E. Keogh, S. Chu, D. Hart and M. Pazzani, An Online Algorithm for Segmenting Time Series. *In: Proceedings of IEEE International Conference on Data Mining,* pp 289-296, (2001).

[6] S. Sripada, E. Reiter, J. Hunter and J. Yu, Generating English Summaries of Time Series Data using the Gricean Maxims. *In Proceedings of KDD-2003*, pp 187 –196, (2.003)

[7] D.Ruth and M. Peroutka, The interactive computer worded forecast. *In Preprints of the 9th International Conference on Interactive Information and Processing Systems for Meteorology, Oceanography, and Hydrology*, pp 321--326, Anaheim, CA, American Meteorological Society, (1993).

[8] E. Goldberg, N. Driedger and R. Kittredge, 'Using Natural-Language Processing to Produce Weather Reports.' *IEEE Expert* **9**:45-53. (1994).

[9] J.Coch., 'Multimeteo: multilingual production of weather forecasts.' *ELRA Newsletter*, **3(2), (**1998).

[10] http://www.fb10-bremen.de/anglistik/langpro/NLG-table/overview-domain.html

[11] S. Sripada, E. Reiter, and L. Hawizy, *Evaluating an NLG system using post-editing.* Technical Report AUCS/TR0402, Department of Computing Science, University of Aberdeen, (2004).

[12] E. Reiter, NLG vs. Templates. *In Proc of the Fifth European Workshop on Natural-Language Generation (ENLGW-1995)*. Leiden, The Netherlands, (1995).

[13] R. Kittredge, and E. Goldberg, and K. Myunghee and A. Polgu`ere, Sublanguage Engineering in the FOG System, *In Proceedings of the Fourth ACL Conference on Applied Natural Language Processing*, Stuttgart, (1994).

[14] J.Coch., Interactive Generation and Knowledge Administration in Multimeteo, *In Proc. Of Content Visualisation and Intermedia Representations (CVIR'98)*, (1998).

[15] R. Milne, The Secrets of Successful Applications, *In Proc. Of Second Singpore International Conference on Intelligent Systems (SPICIS'94)*, Singapore, (1994).

[16] E. Reiter, 'Pipelines and Size Constraints.' *Computational Linguistics.* **26**:251-259, (2000).