Proceedings of the 2013 Winter Simulation Conference
R. Pasupathy, S.-H. Kim, A. Tolk, R. Hill, and M. E. Kuhl, eds

USING A NATURAL LANGUAGE GENERATION APPROACH TO DOCUMENT SIMULATION RESULTS

James Curry
Weihang Zhu
Brian Craig
Lonnie Turpin, Jr.
Majed Bokhari
Pavan Mhasavekar

Industrial Engineering
Lamar University
Beaumont, TX 77710, USA

ABSTRACT

Simulation experiments generate large data sets that must be converted into recommendations for decision makers. This article explores using a Natural Language Generation (NLG) approach for writing summaries of simulation experiments. The article discusses the steps required to convert simulation experiment data to text and highlights the unique aspects of data to text for simulation experiments. Automation of report generation can potentially reduce the time and cost of simulation studies and improve the reporting of results. A prototype software system was developed and applied to a simulation to illustrate the benefits of a NLG approach.

1 INTRODUCTION

This article discusses using Natural Language Generation (NLG) to convert output data from simulation experiments to text. NLG software converts facts to human readable text using a series of steps. NLG has had renewed research interest in part due to several recent efforts in summarizing complex data sets (Portet et al. 2009). Simulation experiments often generate large data sets that must be interpreted and presented to the decision maker. A significant amount of time and effort in simulation studies is spent writing reports and summarizing data. Often only a small subset of simulation results are interesting to decision makers, so an editing process must be applied to the raw data set. Applying NLG to simulation is a promising approach to streamline simulation studies by automating model documentation and output analysis.

Most industrial simulation studies have common analysis goals and steps, but the texts of the reports are different due to the system being studied, goals of the simulation experiment and the results of the simulation experiment. The commonality in many simulation studies facilitates a NLG approach to automatically generating effective reports. To illustrate the NLG approach, this paper examines a simulation study with a predefined experiential design, a single objective cost based on the values of the dependent and independent variables, and a series of constraints based on system performance. This experimental setup occurs in many simulation studies. Similar approaches could be applied to simulation studies that employ optimization to determine the setting for independent variables or studies with multiple objectives.

This paper is organized as follows. Section 2 provides a brief discussion of NLG technology and a discussion of data to text systems. Section 3 discusses the unique aspects of reporting simulation study results. Section 4 presents the NLG steps used in our prototype software and discusses how these steps

can be applied to transform simulation data to text summaries of experiments. Section 5 discusses the software used to develop our prototype system. Section 6 presents a small illustrative case study using the software developed in this project. Section 7 presents conclusions and discusses future research directions.

2 NLG BACKGROUND

NLG technology creates computer systems that present information to users in formats that are easy to comprehend (Reiter and Dale 2007). The NLG approach begins with some nonlinguistic representation of information as input and uses knowledge about language and the application domain to automatically produce documents, reports, explanations, and other forms of texts (Reiter and Dale 2000). NLG systems attempt to develop specialized forms of outputs personalized to suit the receiver. In order to appropriately generate natural-language text, a system must be able to determine what information to include and how to organize this information to achieve its communicative goal (McKeown 1985).

Hunter et al. (2012) used a series of defined steps (data analysis, content determination, aggregation and microplanning, and realization) for converting data to text. Content determination selects and organizes relevant information from the data analysis to be communicated. Aggregation and microplanning convert the information into a series of assembled expression. Finally, in the realization process, the assembled expressions are transformed into a string of text with the correct grammar, word order and punctuation.

A mail merge approach similar to the functionality found in Microsoft Word and other popular document creation packages generates text based on a template document with a few variable fields (Reiter and Dale 1997). The variable fields are typically populated based on information found in a single table or a few tables joined into a single table. A mail merge approach can be effective in generating a form style letter where only a small percentage of the document changes for different recipients. The NLG approach extends the mail merge approach by building the text from raw facts. In NLG systems, a distinction exists between template based systems that directly map nonlinguistic inputs to the linguistic surface and systems that do not use a template format, but the quality of output of template based systems is not always inferior (van Deemter, Krahmer, and Theune 2005). The prototype system developed in this research project has some features of a template based system in that parts of the document are canned text based on a template approach and other sections are not template based. While some text is template based, a realization engine is used to apply grammar rules. Some aggregation of ideas is also used to generate non-repetitive sentences.

The Sweave package combines Latex for typesetting and R for data analysis to generate reports that are dynamic and can be rerun based on the input of new data (Leisch 2002). Integrating data analysis and documentation of study results allows results to be easily reproduced (Leisch 2002). The Sweave mail merge style approach provides an effective framework to combine analysis software with documentation, but does not have built-in tools to aid with grammar.

NLG systems have been deployed in a variety of domains including interactive museum guides (Stock et al. 2007), weather forecasts (Sripada, Reiter, and Davy 1986), gas turbine status reports (Yu et al. 2007), and sensor data (Reddington and Tintarev 2011). Hassan et al. (1997) used natural language generation to document the sample path of an agent representing a person in a social science simulation environment. Their goal was to construct simulated biographies of agents.

Reiter and Dale (2000) noted writing about data requires data analysis and interpretation stages prior to text generation. Data interpretation and the description of the data interpretation are linked tasks and the system design should consider ease of explanation when constructing an experiment. The data analysis step distinguishes data-to-text applications from other kinds of NLG systems whose input tends to consist of information that is already fully structured (Gatt et al. 2009). The output from a simulation experiment under consideration in this paper is highly structured but must be analyzed and summarized be-

fore reporting. All the potential results of the analysis stage must be considered when developing the report.

Several recent research efforts have explored generating text from large complex data sets as a decision-support aids. The Baby Talk project at Aberdeen (Portet et al. 2009) created a set of NLG systems which can generate textual summaries of clinical data about patients in a neonatal intensive care unit. The authors state that with concerted effort, data-to-text technology can improve markedly, to the point where it can help people understand large data sets, not just in medicine but also in engineering, meteorology, finance, and many other areas.

Mahamood and Reiter (2011) in a project related to Baby Talk describe the importance of not only informing, but also taking into consideration the emotional state and knowledge of the recipients when communicating information. For instance, a status update for a parent of a patient would be different than the summary medical note in a chart in terms of detail and tone. This project develops summaries for decision makers that must balance adequately explaining the study with the length of report.

3 REPORTING SIMULATION RESULTS

Communicating simulation results is a key step of the simulation model process and important to the success of the simulation project. Wieland and Nelson (2009) noted that the focus of most simulation software environments is on developing models and analyzing results instead of reporting model results. They also discuss how the lack of standardization in reporting can lead to misinterpretation of findings in the context of confidence intervals and output plots.

Reporting simulation results can be time consuming. Simulation experiments are often expanded based on initial results. Unlike physical experiments, additional data in simulation experiments can be collected at a relatively small cost. While additional experiments can usually be run at a low cost, the analysis and documentation of the experiment adds to cost and overall time required for the project. An effective generated report reduces the effort by inserting the additional or updated results into an accepted reporting format.

Simulation studies are often refined based on expert feedback after preliminary experiments as part of validation and verification. While most model refinement during verification and validation aims to improve the model, refinement can be used to make the model output match the modelers or project sponsors goals. NLG slightly reduces the ability of the modeler to control the interpretation of the study by formally defining the report prior to running the model. This approach has the potential to limit modeler bias in the reported results of a simulation study. While a modeler can and should add interpretation and refinement to the generated text, having a solid initial report generated from the tool would limit the modeler's ability to stray too far from the underlying results.

Sargent (2005) and Colley (1977) discuss the importance of summary and detailed documentation in simulation studies. Reports for decision makers are often formatted into a short report body that states the conclusions of the experiments and longer appendices that provide detailed information about the model and analysis of the results. Our experience with survey report generation project is that decision makers want relatively short reports without excessive statistical analysis. Decision makers also want access to organized detailed information that most likely will not be read or studied in detail. After the main report, relatively long appendixes that provide detailed statistical analysis of all experiments and coding of the model can be provided for interested readers. The writing of the body of the report requires significant editing of findings and interpretation of results that is more challenging than providing all the results in a table or graph format that can be performed by outputting the data to an organized spreadsheet or appendix.

4 NLG STEPS

Identifying a clear communication goal is critical to developing a NLG system. This paper's communication goal is to describe the results of the simulation experiment based on a user specified experimental design. The inputs to the program are independent variables that the user can control, dependent variables

that are the results of the simulation study and scenario variables to demonstrate how the system operates under different conditions. Some examples of independent variables in a simulation experiment are resource levels including number of machines and amount of inventory and control policy settings such as dispatching rules employed. The dependent variables in the simulation are measures of system performance based on the simulation experiment such as number of sales, revenue, average wait time, flow time, number of late jobs and tardiness. The scenario variables define alternative operating conditions such as different arrival rates and cost parameters.

The program converts this information into a report using additional meta data about the independent and dependent variables. The meta data contains a description of the fields for the report. The meta data defines costs for the independent and dependent variables, so that the variables can be converted into a single number for total cost or profit. The meta data also indicates if the field is a numeric field or a categorical policy. The meta data file also defines upper and lower bound constraints for the dependent variables based on user's goals. Each scenario can have different costs associated with and limits on variables. This flexibility allows for a wide range of scenario definitions to be supported by the software. The meta data allows a report to be constructed for a simulation where the goal is to minimize cost or maximize profit subject to limits on dependent variables.

The current prototype focuses on paragraphs instead of bullet lists or tables of simulation finding. Bulleted lists and tables can also be an effective approach for succinctly explaining simulation results. The same approached could be applied to generated text formatted as bulleted lists or text to support tables of data.

NLG systems differ from mail merge systems in part by having organized steps for going from data to text. A mail merge approach primarily uses fixed text and inserts fields with variable information into the document. This project has a series of steps for data analysis, content determination, aggregation and microplanning, and realization that are similar to Hunter et al. (2012). The following subsections discuss these steps in the context of describing the results of a simulation experiment.

4.1 Data Analysis

The data analysis step first determines the feasible configurations for each scenario. A solution is feasible if all dependent variables confidence limits are within the user specified acceptable values. The feasible and infeasible solutions are ranked based on cost of independent and dependent variables in the simulation run. The ranked feasible and infeasible solutions are sent to content determination to identify the results to be included in the summary of the simulation experiment.

4.2 Content Determination

Content determination identifies the facts to be presented to the user. The system must establish what simulation results should be presented to the user. The two primary content elements in our project are an explanation of the experimental design and a discussion of the results. Our prototype employs simple rules for selecting information interesting to the user. The lowest cost alternative for a scenario that satisfies users requirements is presented. The software has simple rules that specify when to discuss infeasible solutions and feasible solutions with cost above the lowest cost feasible solution.

4.3 Microplanning and Aggregation

Aggregation combines multiple concepts into a single sentence to avoid repetitive sentences. This step is critical when reporting simulation results and describing simulation experiments. As an illustration, the following text is not natural:

> Scenario one has an arrival rate of 1 job per hour. Scenario two has an arrival rate of 2 job per hour. Scenario three has an arrival rate of 3 job per hour. Scenario four has an arrival rate of 4 job per hour.

A more natural wording of the sentence would be the four scenarios explored arrival rates from 1 to 4 jobs per hour. Aggregation results across scenarios is also important for readability if the best setting for input variables is consistent across many or all scenarios.

Our prototype implementation makes several key word choice decisions. Based on settings, the wording of the document can be in terms of maximizing profit or minimizing cost. Other word choice decisions based on the data describe the difference between the best setting for the independent variables and other alternatives for a given scenario. When a variable represents a policy, a text description of the policy is used to describe it as opposed to a numeric representation of the variable value. Settings also determine if and how the confidence limits of the simulation experiment are displayed to the user and how much discussion of the simulation results is given.

4.4 Realization

Realization applies grammar rule to join words into sentences correctly (Gatt and Reiter 2009). The key elements of realization are selecting the word forms such as plural and verb form (morphology), capitalization and punctuation (orthography). Realization also considers word order based on the specified structure of the sentence.

This project selected SimpleNLG as the realizer software. SimpleNLG is a Java API for realization engine that was designed to be easy to use for researchers focusing on tasks other than realization when constructing NLG application (Gatt and Reiter 2009). SimpleNLG is open source software available under Mozilla Public License 1.1. SimpleNLG organizes the sentence, selects inflected forms of words for number of items and tense, manages punctuation, and handles lists (Gatt and Reiter 2009). Without a realization engine, these tasks would be overwhelming when generating text. As a simple illustration, consider writing the best found solution for resource levels for a particular scenario given an array *resource_names* that stores the name of each resource type, an array *number_of_resources* that stores the number of resources required by type, and a string *scenario*. From this information, an author could describe the lowest cost alternative in a sentence as follows:

> The lowest cost resource configuration for scenario two is three lathes, four mills and a drill.

Developing a computer program to write the above sentence would be very difficult without using or writing a custom realization engine. To make nouns plural in the object phase, the developer could either ask the user to enter the plural names of items into the system or store a dictionary of the English language. Organizing a list of items into a sentence is a relatively easy task but cumbersome using a series of if statements and loops. With SimpleNLG engine, the following Java code (Figure 1) generates the above sentence based on the *resource_names, number_of_resources, and scenario* variables.

```
SPhraseSpec p = nlgFactory.createClause();  // create the sentence object.
CoordinatedPhraseElement object_phrase = nlgFactory.createCoordinatedPhrase();
for(int i = 0 ; i<resource_names.length; i++) // For all resources
    {
        NPPhraseSpec r = nlgFactory.createNounPhrase(resource_names[i]);
        if(number_of_resources[i] > 1 ) //Make noun plural if more than one resource is used
        { // Use plural form of resource name with the number of resources in front.
            r.setPlural(true);
            r.addPreModifier(number_to_text(number_of_resources[i]));
        }
    else
        { //Use single form with a in front.
            r.addPreModifier("a");
        }
    object_phrase.addCoordinate(r);
    }
p.setSubject("the lowest cost resource configuration for scenario " + number_to_text(scenario));
p.setObject(object_phrase);
p.setVerb("is");
output = realiser.realiseSentence(p);
```

Figure 1: Illustrative Java code to describe a scenario.

5 SOFTWARE FRAMEWORK

This project uses R for developing text summaries of the simulation results. The R program calls SimpleNLG version 4.4 for realization via rJava package (Urbanek 2011). The R language was selected due to the ease of data analysis and selecting subsets of information from tables stored as data frames.

The data from the simulation is transformed into a demoralized table to describe the independent, dependent and scenario variables in each simulation replication. The table contains the replication ID, the name of the variable, the value of the variable, the limits on the variable, the cost of the variable, the text to describe the variable for non-numeric variables, run length of simulation and the unit of measure for numeric variables. This basic input is converted into text via a series of R functions.

The prototype has several user settings that control the format and information in the output report:

1. Writing the report in terms of minimizing cost or maximizing profit.
2. Amount of improvement in dependent variables that justifies discussing a solution that costs more than the lowest cost solution.
3. The number of alternative feasible solutions with costs greater than the lowest cost solution to include in the report.
4. How much infeasibility to allow in an infeasible alternative that is included in the report.
5. The number of infeasible solutions to include in the report.
6. How much detail to provide when discussing the cost calculations.
7. How to display numeric results from the simulation in terms of number format and confidence limits.

User settings allow the report to be customized. Our prototype uses simple rules to determine what solutions to discuss beyond the lowest cost alternative, since content determination was not the focus of the initial phase of this project. While these rules are effective for simulations with few feasible alternatives, complex simulations with a large number of similar solutions would require more sophisticated approaches to select on a small subset of solutions to discuss.

A simulation was constructed in Rockwell Arena Version 14 to generate raw simulation data. The simulation was run with the Arena Process Analyzer. The output of Process Analyzer is combined with meta data and sent to a flat file to be read by the R report generation prototype.

6 CASE STUDIES

This article presents a simple case study to illustrate the NLG prototype system developed. The case study is a simple simulation to determine the number of machines required for a manufacturing system. The study determines the number of parallel machines required under different demand rates. The demand rates in the scenarios are 5 jobs per hour, 10 jobs per hour, 15 jobs per hour, and 20 jobs per hour. The dependent variables in the simulation experiment is flow time. The average flow time for jobs is restricted to less than 1 hour. The processing time for jobs follows a uniform distribution from 7 to 15 minutes. The number of machines ranged from 1 to 6. Each machine costs $500,000. Each configuration is run for 20 replications that are 10,000 hours long with a 1,000 hour warm up period. Jobs are scheduled first come first served. The results of the simulation experiment are displayed in Table 1.

Table 1: Results of simulation experiment in table format.

Arrival Rate (Jobs Per Hour)	Number of Machines	Flow Time (Minutes)
5	1	76.0
5	2	12.6
5	3	11.2
5	4	11.0
5	5	11.0
5	6	11.0
10	2	41.0
10	3	12.9
10	4	11.4
10	5	11.1
10	6	11.0
15	3	30.9
15	4	13.0
15	5	11.5
15	6	11.2
20	4	25.6
20	5	13.0
20	6	11.6

The output from the simulation experiment is sent to the R code for producing a text summary. The summary is based on minimizing capital cost given the constraint that flow time is less than one hour. On-

ly interesting results are presented. User settings define interesting results to discuss beyond the optimal solution. Infeasible results are considered interesting if they are within 50% of the constraint bounds for dependent variables and the cost is reduced. Feasible results beyond the lowest cost alternative are considered interesting if the lowest cost solution is close to the constraint as defined by being within 50% of the constraint and the solution is an improvement for the dependent variables. These settings result in the text summary of the simulation experiment displayed in Figure 2.

> This study examines four scenarios. The four scenarios have different arrival rates from five to twenty jobs per hour. The goal of the study is to find the lowest cost systems that satisfies a user defined constraint of flow time less than 60 minutes. The simulation explored using 1 to 6 machines that cost $500,000 each. Each simulation was run for twenty replications with replication length 10,000 hours and warm up period 1,000 hours.
>
> Scenario one explores an arrival rate of five jobs per hour. The best feasible solution found in the simulation experiment has two machines at a cost of $1,000,000. The flow time in the simulation was 13 minutes that is significantly under the constraint of 60 minutes. A configuration with one machines is an infeasible alterative with flow time of 76 minutes that costs $500,000.
>
> Scenario two explores an arrival rate of ten jobs per hour. The best feasible solution found in the simulation experiment has two machines at a cost of $1,000,000. The flow time in the simulation was 41 minutes that is under the constraint of 60 minutes. A configuration with three machines is an alternative solution with flow time of 13 minutes that costs $1,500,000.
>
> Scenario three explores an arrival rate of fifteen jobs per hour. The best feasible solution found in the simulation experiment has three machines at a cost of $1,500,000. The flow time in the simulation was 31 minutes that is under the constraint of 60 minutes. A configuration with four machines is an alternative solution with flow time of 13 minutes that costs $2,000,000.
>
> Scenario four explores an arrival rate of twenty jobs per hour. The best feasible solution found in the simulation experiment has four machines at a cost of $2,000,000. The flow time in the simulation was 26 minutes that is significantly under the constraint of 60 minutes.

Figure 2: Text generated to describe the simulation experiment.

7 CONCLUSION AND FUTURE RESEARCH

NLG can be an effective tool for building automated reports of simulation systems. Using a realization engine can significantly reduce the amount of programming required to develop generic simulation reports. The combination of two open source software tools, R and SimpleNLG, is an effective development environment for simulation data to report text. A key challenge in developing generic simulation reports are identifying what information is useful to the decision maker. Our initial prototype employed a simple rule based approach to determine when to display feasible and infeasible solutions beyond the lowest cost feasible configuration.

The NLG approach discussed in this paper has the potential for improving how simulation studies are executed. By reducing the effort required to report results, modelers can focus on modeling and experimental design instead of documentation. Being able to quickly regenerate a report with additional scenarios and design alternatives would reduce the time and cost required to modify a simulation experiment.

Several future research exist for this project. Analysis of simulation differs slightly from most experimental data in that additional results are relatively inexpensive to produce. As a result, a generated report that can be easily recreated could be an effective tool. This project developed a simple prototype to illustrate the effectiveness of NLG techniques to achieve this goal. The prototype developed can be expanded and improved to be effective in more simulation environments with additional output reports and formats including detailed report documents and high level slides for presentations. We are currently working on an effective IDE for our current prototype that combines experimental design, output formatting and reporting into a single interface. We are also expanding the tool to produce summaries of optimization based simulation experiments.

Another promising direction is to use NLG to document the simulation model. Simulation models are often complex and difficult to explain to decision makers who do not understand the simulation language. Documenting entity paths within the simulation model for testing and model documentation could be aided with a NLG approach.

Content determination especially summarizing large data sets in short summaries is a NLG research area that is especially important for effective reporting of simulation experiments. Simulation can generate vast data sets based on running an experimental design or optimization. Selecting a subset of information to display to the user can be a major task for modelers when writing simulation reports. Alternative solutions can be presented in text format, table format, or summaries based on statistical models such as linear regression. Given that simulations can generate thousands of feasible solution, presenting a small number of diverse high quality representative solutions beyond the best solution can explain the solution space to decision makers. Content determination can be viewed as an optimization problem of maximizing the amount of information in the report given page length constraints.

REFERENCES

Colley, B. 1977. "Documenting Simulation Studies for Management." In *Proceedings of the 1977 Winter Simulation Conference,* edited by H. J. Highland, 743-746. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Gatt, A., and E. Reiter. 2009. "SimpleNLG: A Realisation Engine for Practical Applications." In *Proceedings of the 12th European Workshop on Natural Language Generation*, edited by E. Krahmer and M. Theune, 90-93. Association for Computational Linguistics.

Gatt, A., F. Portet, E. Reiter, J. Hunter, S. Mahamood, W. Moncur, and S. Sripada. 2009. "From Data to Text in the Neonatal Intensive Care Unit: Using NLG Technology for Decision Support and Information Management." *Ai Communications* 22(3):153-186.

Hassan, S., J. Pavón, M. Arroyo, and C. Leon. 2007. "Agent Based Simulation Framework for Quantitative and Qualitative Social Research: Statistics and Natural Language Generation." In *Proceedings of the Fourth Conference of the European Social Simulation Association*, edited by F. Amblard, 697-707.

Hunter, J., Y. Freer, A. Gatt, E. Reiter, S. Sripada, and C. Sykes. 2012. "Automatic Generation of Natural Language Nursing Shift Summaries in Neonatal Intensive Care: BT-Nurse." *Artificial intelligence in medicine* 56(3): 157-172.

Leisch, F. 2002. "Sweave. Dynamic Generation of Statistical Reports Using Literate Data Analysis." Report Series SFB Adaptive Information Systems and Modelling in Economics and Management Science, WU Vienna University of Economics and Business, Vienna.

Mahamood, S., and E. Reiter. 2011. "Generating Affective Natural Language for Parents of Neonatal Infants." In *Proceedings of the 13th European Workshop on Natural Language Generation*, edited by C. Gardent and K. Striegnitz, 12-21. Association for Computational Linguistics.

McKeown, K. R. 1985. "Discourse Strategies for Generating Natural-Language Text." *Artificial Intelligence* 27(1):1-41.

Portet, F., E. Reiter, A. Gatt, J. Hunter, S. Sripada, Y. Freer, and C. Sykes. 2009. "Automatic Generation of Textual Summaries from Neonatal Intensive Care Data." *Artificial Intelligence* 173(7):789-816.

Reddington, J., and N. Tintarev. 2011. "Automatically Generating Stories from Sensor Data." In *Proceedings of the 16th International Conference on Intelligent User Interfaces*, edited by P. Pu and M. Pazzani, 407-410. ACM.

Reiter, E., and R. Dale. 1997. "Building Applied Natural Language Generation Systems." *Natural Language Engineering* 3(1):57-87.

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

Sargent, R. 2005. "Verification and Validation of Simulation Models." In *Proceedings of the 2005 Winter Simulation Conference*, edited by M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, 130-143. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

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

Stock, O., M. Zancanaro, P. Busetta, C. Callaway, A. Krüger, M. Kruppa, T. Kuflik, E. Not, and C. Rocchi. 2007. "Adaptive, Intelligent Presentation of Information for the Museum Visitor in PEACH." *User Modeling and User-Adapted Interaction* 17(3):257-304.

Urbanek, S. 2011. *rJava: Low-level R to Java Interface.* R Package Version 0.9-3.

van Deemter, K., E. Krahmer, and M. Theune. 2005. "Real Versus Template-based Natural Language Generation: a False Opposition?" *Computational Linguistics* 31(1):15-24.

Wieland, J. R. and B. Nelson. 2009. "How Simulation Languages Should Report Results: a Modest Proposal." In *the Proceedings of the 2009 Winter Simulation Conference*, edited by M. D. Rossetti, R. R. Hill, B. Johansson, A. Dunkin, and R. G. Ingalls, 709-715. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Yu, J., E. Reiter, J. Hunter, and C. Mellish. 2007. "Choosing the Content of Textual Summaries of Large Time-series Data Sets." *Natural Language Engineering* 13(1): 25-50.

AUTHOR BIOGRAPHIES

JAMES CURRY is an Associate Professor in the Industrial Engineering Department at Lamar University. His research interests are simulation, optimization, data mining and natural language generation. His email and web address is james.curry@lamar.edu and dept.lamar.edu/industrial.

WEIHANG ZHU is an Associate Professor in the Industrial Engineering Department at Lamar University. His research interests are information technology, optimization, simulation and human computer interfaces. His email and web address is weihang.zhu@lamar.edu and maritime.lamar.edu/personal/zhu.

BRIAN CRAIG is a Professor and Chair of the Industrial Engineering Department at Lamar University. His research interests are human factors, ergonomics and safety. His email and web address is brian.craig@lamar.edu and dept.lamar.edu/industrial.

LONNIE TURPIN, JR. is an doctoral student in the Industrial Engineering Department at Lamar University. He is currently employed as an Instructor in the Department of Management, Marketing and

Business Administration at McNeese State University. His research interests are Operations Research and Decision Analysis. His email address is lturpin@mcneese.edu.

MAJED BOKHARI is a doctoral student in the Industrial Engineering Department at Lamar University. His research interests are optimization and data mining. His email and web address is mbokhari@my.lamar.edu and dept.lamar.edu/industrial.

PAVAN MHASAVEKAR is a doctoral student in the Industrial Engineering Department at Lamar University. His research interests are lean manufacturing, inventory control and simulation. His email and web address is prmhasaveka@lamar.edu and dept.lamar.edu/industrial.