Automated Briefing Production for Lessons Learned Systems

Inderjeet Mani and **Kristian Concepcion** and **Linda Van Guilder**
The MITRE Corporation, W640
11493 Sunset Hills Road
Reston, Virginia 22090
{imani,kjc9, lcvg}@mitre.org

Abstract

Document production is an important function in many organizations. In addition to instruction manuals, courseware, reports, system documentation, etc., briefings are a very common type of document product, often used in slide form as a visual accompaniment to a talk. This paper describes a domain-independent system for automatic briefing generation from a high-level outline provided by the user. We discuss how some of the methodologies used in this system may be leveraged in intelligent lessons learned systems.

Introduction

Document production is an important function in many organizations. In addition to instruction manuals, courseware, reports, system documentation, etc., briefings are a very common type of document product, often used in slide form as a visual accompaniment to a talk. Since so much time is spent by so many people in producing briefings, often under serious time constraints, any method to reduce the amount of time spent on briefing production could yield great gains in productivity. In this paper, we provide an overview of a briefing generation system, followed by a brief discussion as to how some of the methodologies used in this system may be leveraged in intelligent lessons learned systems.

Many briefings have a stereotypical structure, dictated in part by the business rules of the organization. For example, a commander may present a daily or weekly brief to her superiors, which is more in the nature of a routine update of activities since the last briefing; or she may provide an action brief, which is triggered by a particular situation, and which consists of a situation update followed by arguments recommending a particular course of action. Further, the process of constructing a briefing may involve certain stereotypical activities, including culling information from particular sources, such as messages, news, web pages, previous briefings, etc. Thus, while part of the briefing content may be created anew by the briefing author, other parts of the briefing may be constructed from existing information sources. However, information in those sources need not necessarily be in the same form as needed by the briefing.

Our work forms part of a larger DARPA-funded project (Project Genoa) aimed at improving analysis and decision-making in crisis situations by providing tools tha t allow analysts to collaborate to develop structured arguments in support of particular conclusions and to help predict likely future scenarios. These tools leverage a corporate memory that acts as a repository of knowledge gained as a result of analytical tasks. The arguments, along with background evidence, are packaged together as briefings to high-level decision-makers.

Our approach needs to allow the analyst to take on as much of the briefing authoring as she wants to (e.g., it may take time for her to adapt to or trust the machine, or she may want the machine to present just part of the briefing). The analyst's organization usually will instantiate one of several templates dictating the high-level structure of a briefing; for example, a briefing may always have to begin with an executive summary. The methods used also need to be relatively domain-independent, since the subject matter of crises are somewhat unpredictable.

Given these task requirements, we have adopted an approach that is flexible about accommodating different degrees of author involvement, that is relatively neutral about the rhetorical theory underlying the briefing structure (since a template may be provided by others), and that is domain-independent. In our approach, the author creates the briefing outline, which is then fleshed out further by the system based on information in the outline. The system fills out some content by invoking information reduction tools called summarization filters; it also makes decisions, when needed, about output media type; it introduces narrative elements to improve the coherence of the briefing; and finally, it assembles the final presentation, making decisions about spatial layout in the process.

A briefing is represented as a tree. The structure of the tree represents the rhetorical structure of the briefing. Each node has a label, which offers a brief textual description of the node. Each leaf node has an associated goal, which, when realized, provides content for that node. There are two kinds of goals: content-level goals and narrative-level goals. Content-level goals are also of two kinds: *retrieve* goals, which retrieve existing media objects of a particular type (text, audio, image, audio, video) from corporate memory satisfying some description, and *create* goals, which create new media objects of these types using programs. The programs we have used for create goals are summarization filters, which take input information and turn it into some more abstract and useful representation tailored to the needs of the end-user, filtering out unimportant information.

Narrative-level goals introduce descriptions of content at other nodes: they include captions and running text for media objects, and segues, which are rhetorical moves describing a transition to a node. Ordering relations reflecting temporal and spatial layout are defined on nodes in the tree. Two coarse-grained relations, *seq* for precedence, and *par* for simultaneity, are used to specify a temporal ordering on the nodes in the tree. The tree representation, along with the temporal constraints, can be rendered in text as XML; we refer to the XML representation as a script.

The overall architecture of our system is now described. The user creates the briefing outline in the form of a script, by using a GUI. The briefing generator takes the script as input. The Script Validator applies an XML parser to the script, to check for syntactic correctness. It then builds a tree representation for the script, which represents the briefing outline, with temporal constraints attached to the leaves of the tree.

Next, a Content Creator takes the input tree and expands it by introducing narrative-level goal cover segues to nodes, and running text and captions describing media objects at content nodes. Segues are generated based on node labels, distance between nodes, and the height in the tree. Running text and short captions are generated from meta-information associated with media objects. For example, in the case of a create goal, a summarization filter may take in a collection of documents and produce a graph of some trend detected in the collection, say, an association between people and organizations over time. Or else, it may generate a biography of a person from the collection. The meta-information in the structured representation of these outputs is used to generate the running text and captions, by using shallow text generation methods (canned text). Where the media objects are found by retrieve goals, the system relies on existing meta-information, but is designed not to fail if there isn't any - i.e., it will fail to provide running text.

The end result of content selection (which has an XML representation called a ground script) is that the complete tree has been fully specified, with all the create and retrieve goals fully specified, with all the output media types decided. When summarization filters are used (for create goals), the media type of the output is specified as a parameter to the filter. This media type may be converted to some other type by the system (e.g., text to speech conversion using Festival (Taylor et al. 98). By default, all narrative nodes attempt to realize their goals as a speech media type, using rules based on text length and truncatability to decide when to use text-to-speech. Captions are always realized, in addition, as text (i.e., they have a text realization and a possible audio realization).

Then, a Content Executor executes all the create and retrieve goals. This is a very simple step, resulting in the generation of all the media objects in the presentation, except for the audio files for speech to be synthesized. Thus, this step results in realization of the content at the leaves of the tree.

Finally, the Presentation Generator takes the tree which is output from Content Execution, along with its temporal ordering constraints, and generates the spatial layout of the presentation. If no spatial layout constraints are specified (the default is to not specify these), the system allocates space using a simple method based on the temporal layout for nodes which have spatial manifestations. Speech synthesis is also carried out here. Coherence of a presented segment is influenced mainly by the temporal constraints (which have been fleshed out by the Content Creator to include narrative nodes). Once the tree is augmented with spatial layout constraints, it is translated by the Presentation Generator into SMIL1 (Synchronized Multimedia Integration Language)(SMIL 99), a W3C-developed extension of HTML that can be played by standard multimedia players (such as (REAL Player 99) and (Grins Player 99). This step thus presents the realized content, synthesizing it into a multimedia presentation laid out spatially and temporally.

Relevance to Intelligent Lessons Learned Systems

Although the briefing generation system was not designed specifically for Lessons Learned, the tools we have developed use methodologies that are relevant to Lessons Learned.

The Genoa briefing module is aimed at packaging arguments in favor of a particular analysis, which is presented along with supporting evidence to decision makers. In such cases, the briefing outline can specify arguments and their supporting evidence (which is filled in and reinforced by create and retrieve goals, both of which draw and build from data in corporate memory). Representations such as (Toulmin 58) structures have been used in a variety of AI domains to convey arguments (Clark 91) (Janssen and Sage 96). Toulmin structures identify the claim (the conclusion reached), its grounds (facts on which the argument is based), its warrant (the inference rules applied to arrive at the conclusion), its qualifier (degree of certainty), as well as its backing (justification) and a possible rebuttal (exceptions that invalidate the claim).

These kinds of explanations can be very useful in lessons learned as well. For example, the lesson learned, the experience which gave rise to it, the impact it has on task performance, and the specific process that it corrects or reinforces can be presented in terms of arguments. The briefing generation module can help in presenting the lessons learned, by culling information related to the arguments and their supporting evidence from web and other sources, which can be fleshed out by the retrieval and creation goals. Thus, canonical presentations of explanations related to lessons learned can be carried out. Further, where the lessons learned are fairly complex, summaries of these lessons can be presented, allowing a drill-down capability as needed.

Future Directions

There is a fair amount of work on automatic authoring of multimedia presentations, e.g., (Wahlster et al. 92), (Dalal et al. 96), (Mittal et al. 95), (Andre and Rist 97), generation of documents from specification knowledge bases (Power and Scott, 98), and explanatory caption generation (Mittal et al. 95). These efforts differ from ours in two ways: first, unlike us, they are not open-domain; and, second, they don't use summarization filters. The open-domain requirement of Project Genoa has required the design of a generic method. However, in many domains, especially for Lessons Learned systems, domain knowledge bases may be available which can be exploited by our approach to allow for more reasoning about the content of the presentation. This can be viewed in part as an opportunity to leverage much richer meta-info specifications than the ones we currently use, including representations of evidence and counter-evidence for an argument or point-of-view. Further, more effective tailoring of presentations to individual users or students may be exploited, based on intensive domain modeling (Daniel et al. 99). Of course, we will need to carry out an evaluation to assess the extent to which the automation described here provides efficiency gains in document production.

A second direction for future work is the output format. While multimedia players are popular, most presentations we have come across are in Microsoft Powerpoint. In addition, if the user wants to edit the output, Powerpoint editing is preferred to having to use a SMIL (i.e., HTML) editor. We are therefore investigating the option of producing Powerpoint presentations.

Finally, we have begun the design of the Script Creator GUI (the only component in Figure 1 remaining to be built). This will allow the author to create scripts for the briefing generator (instead of editing templates by hand), by laying out icons for media objects in temporal order (seq of pars). A user will be able to select a "standard" briefing template from a menu, and then view it in a briefing/template structure editor. The user can then provide content by adding annotations to any node in the briefing template. The user has a choice of saving the edit version in template forma, or in a particular output format e.g., (Powerpoint, SMIL).

References

Andre, E. and Rist, T. 1997. Towards a New Generation of Hypermedia Systems: Extending Automated Presentation Design for Hypermedia. L. Dybkjaer, ed., Proceedings of the Third Spoken Dialogue and Discourse Workshop, Topics in Natural Interactive Systems 1. The Maersk Mc-Kinney Moller Institute for Production Technology, Odense University, Denmark, 10-27.

Clark, P. 1991. A Model of Argumentation and Its Application in a Cooperative Expert System. Ph.D. Thesis, Turing Institute, University of Strathclyde, 1991.

Dalal, M., Feiner, S., McKeown, K., Pan, S., Zhou, M., Hollerer, T., Shaw, J., Feng, Y., and Fromer, J. 1996. Negotiation for Automated Generation of Temporal MultimediaPresentations. Proceedings of ACM Multimedia '96.

Daniel, B. H., Bares, W. H., Callaway, C. B., Lester, J. C. 1999. Student-Sensitive Multimodal Explanation Generation for 3D Learning Environments. Proceedings of AAAI/IAAI 1999, 114-120.

Grins Player. http://www.oratrix.com

Janssen, T., and Sage, A. P. 1996. Group Decision Support Using Toulmin Argument Structures, IEEE International Conference on Systems, Man, and Cybernetics, Beijing, China, 2704-2709.

Mittal, V., Roth, S., Moore, J., Mattis, J., and Carenini, G. 1995. Generating Explanatory Captions for Information Graphics. Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'95), 1276-1283.

Power, R., and Scott, D. 1998. Multilingual Authoring using Feedback Texts. Proceedings of COLING'98, Montreal, 1053-1059, August 10-14.

Real Player. http://www.real.com

SMIL. http://www.w3.org/AudioVideo/

Taylor, P., Black, A., and Caley, R. 1998. The architecture of the Festival Speech Synthesis System. Proceedings of the Third ESCA Workshop on Speech Synthesis, Jenolan Caves, Australia, 147-151.

Toulmin, S. 1958. The Uses of Argument. University Press, Cambridge.

Wahlster, W., Andre, E., Finkler, W., Profitlich, H-J., and Rist, T. 1993. Plan-Based Integration of Natural Language and Graphics Generation. AI Journal, 63.