Capture, Storage and Reuse of Lessons about Information Resources: Supporting Task-Based Information Search*

David B. Leake, Travis Bauer, Ana Maguitman and David C. Wilson
Computer Science Department
Lindley Hall, Indiana University
150 S. Woodlawn Avenue
Bloomington, IN 47405, U.S.A.
{leake,trbauer,anmaguit,davwils}@cs.indiana.edu

Abstract

Learning how to find relevant information sources is an important part of solving novel problems and mastering new domains. This paper introduces work on developing a lessons learned system that supports task-driven research by (1) automatically storing cases recording which information resources researchers consult during their decision-making; (2) using these cases to proactively suggest information resources to consult in similar future task contexts; and (3) augmenting existing information resources by providing tools to support users in elucidating and capturing records of useful information that they have found, for future reuse. Our approach integrates aspects of case-based reasoning, "just-in-time" task-based information retrieval, and concept mapping. We describe the motivations for this work and how lessons learned systems for suggesting research resources complement those that store task solutions. We present an initial system implementation that illustrates the desired properties, and close with a discussion of the primary questions and open issues to address.

Introduction

Many applied case-based reasoning systems have been developed to capture experiential knowledge in order to aid future decision-making (e.g., (Watson 1997)). Thus case-based approaches provide a technology for *lessons learned* systems (e.g., (Sary & Mackey 1996; Bagg 1997; Weber *et al.* 2000). Typically, case-based lessons learned systems support knowledge reuse by capturing solutions for prior problems and providing them as starting points for future problem-solving. When prior solutions are difficult to adapt, or when new problems are beyond the scope of the case base, expert advice may be needed. For example, many such problems may arise in the early stages of dealing with a new task domain. In this situation, the experts themselves may need to seek information from additional sources, engaging in research to deal with novel problems or to confirm and support their decision making. Even if the problems are initially outside the scope of those experts' knowledge, part of their expertise is "knowing where to look" to find answers: They either know good resources or know efficient research paths to find appropriate resources—an ability novices may not share. For truly novel problems, even experts may need assistance in finding relevant resources. Thus we believe that case-based lessons learned systems for guiding the research process have the potential to significantly aid in finding needed information, both for novices and for experts. Such lessons learned systems differ from typical case-based aiding systems—which capture only the results of task reasoning—in learning lessons about how to perform the supporting research necessary to address novel problems.

This paper describes initial research on the CALVIN project (Context Accumulation for Learning Varied Informative Notes), which investigates how lessons learned systems can support the process of finding information relevant to a particular task. The user of such a system may have any of a wide range of domain-level tasks, such as deciding the likely outcomes of different courses of action, generating feasibility studies, generating or refining designs, etc. These tasks give rise to information needs, which the user must satisfy through research. CALVIN captures lessons about where and how to find information relevant to the user's decision-making task. These lessons are used to provide future users with suggestions of relevant information resources (e.g., web pages, documents, people who have researched similar problems, etc.) The project aims to develop a "resource suggester" system for capturing and proactively providing task-relevant information, based on monitoring a user's research process. The intent is for this resource suggester to learn and provide useful task-driven lessons that aid in navigation of heterogeneous information sources.

Of course, lessons about which information resources

*This research is supported in part by NASA under award No NCC 2-1035. The authors gratefully acknowledge the contributions to this project of Mary Livingston and John Melton at NASA Ames Research Center, and of Alberto Cañas and the Concept Mapping group at the IHMC, University of West Florida. They would also like to thank Dunja Mladenic for useful ideas for the development of the proxy server used in this work.

to consult are only part of what a user needs. Lessons learned systems that guide selection of information sources, like CALVIN, are complementary to lessons learned systems capturing domain content such as solutions to particular problems, and both types of systems can play important roles. Consequently, we are also exploring how to capture useful domain information as resources are consulted. To this end, CALVIN includes the capability for users to add textual annotations to resources describing important points, and uses concept mapping tools to provide a sketchpad for elucidating the user's growing knowledge of the domain as research progresses. These concept maps provide an additional type of information resource for CALVIN to suggest in similar contexts. Thus CALVIN stores both lessons about how to learn information, and records of the domain lessons learned.

Motivations and System Principles

CALVIN is being tested in the task domain of aerospace design, to capture lessons that are useful when expert designers must deal with novel problems that require additional research (e.g., to decide the feasibility of manufacturing a particular wing shape, or whether performance or cost would be increased by including a particular feature in the design of a wing). Expert designers acquire considerable knowledge about relevant sources of information, and develop rich domain models. However, serious problems arise from "knowledge loss" as engineers retire or are transferred to new areas. In discussions at NASA, we learned that especially as designers are moved into new task areas outside their original specialties (e.g., from aircraft design to the design of space vehicles), they may have to expend considerable effort to master the new areas and find the answers they need. Consequently, systems to help their research process and profit from others' research experience could play a significant role. The CALVIN system captures records of the resources that are useful in a given context, to facilitate future research by the same or other designers.

CALVIN's design is shaped by the principles that lessons learned systems should be integrated into the task process, should learn both from monitoring user task performance and by optional user-initiated learning, and should monitor the user's task processing to provide "just in time" retrieval (Leake *et al.* 1999) to provide information automatically as it is needed during the decision-maker's task. These principles are reflected in the following three primary system tasks:

1. Learn about information sources for research: Automatically record cases representing the sources consulted in particular task contexts
2. Learn about the domain: Support annotation of information resources, and provide tools for incremental aggregation, refinement, and capture of records of domain lessons learned from those sources.
3. Proactively suggest relevant information resources: As the user browses, automatically present useful information resources from similar prior contexts

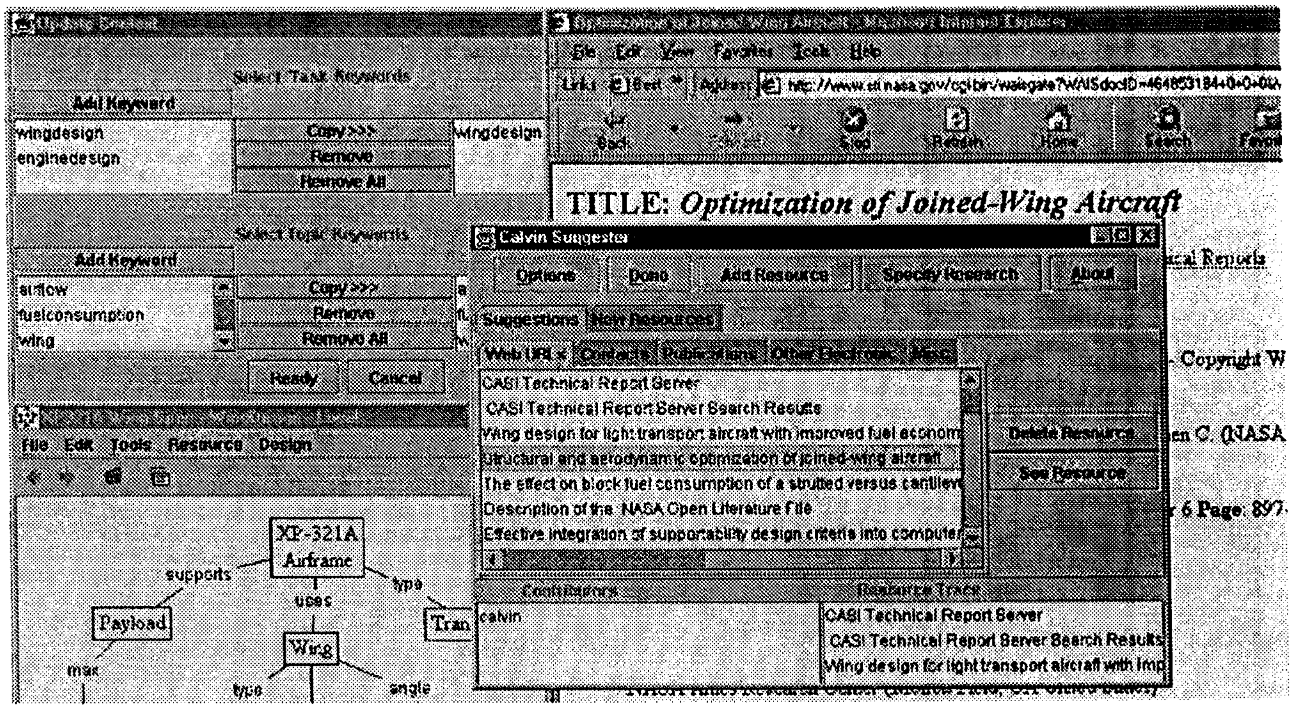
How the System Works

Ideally, a just-in-time information retrieval system will monitor the process the user follows in his or her usual task process, and, from that monitoring, will infer the user's information needs. The research task supported by CALVIN could involve a number of different software tools, such as browsers, search engines, databases, electronic mail, etc., as well as telephone or face-to-face contact. Rather than attempt to provide specialized support for all of these initially, the current system directly monitors only web browsers, but provides capabilities for users to record information about other searches, or resources, and their results (for example, descriptions of books or articles that are available off-line, or about research paths taken to find useful contact people). This information is stored in "resource cases," which are indexed by the task and current search context to be retrieved automatically in similar future contexts. To guard against storing bad choices such as dead-end paths, an editor is provided to allow the user to decide which resources and paths to retain at the end of a research session.

When a system run is started, the user may simply begin browsing, or may choose to generate a description of the user's task and specific topics, to provide an initial context (these terms may be selected from a menu with a pre-defined vocabulary, or added manually). As the user browses pages using a standard browser, the system stores cases representing the content of the pages and the contexts in which they were considered. The system generates a simple description of the page content (without natural language processing) by extracting keywords from the pages visited and combining those keywords with the previously-generated context to suggest resources that were useful in similar prior circumstances.

The user may select any of those pages to see the sequence of pages followed to reach it. This sequence can help the user find additional relevant pages by following a similar sequence after adjusting it for current needs (e.g., using the same specialized search engine that a previous expert consulted to find the current page, but to check for papers on a different topic). The motivation for showing these paths is to assist a simple manual form of derivational analogy.

The system records which resources were consulted, for future retrieval, as well as a trace of the path of pages followed to find the information. (To avoid saving off-topic searches, the user can toggle system monitoring.) When the system detects a similar context in future processing, it suggests these resources to the user. A screen image including the current Suggester display is shown in figure 1.



CASI Technical Report Server
CASI Technical Report Server Search Results
Wing design for light transport aircraft with improved fuel econom
Structural and aerodynamic optimization of joined-wing aircraft
The effect on block fuel consumption of a strutted versus cantilev
Description of the NASA Open Literature File
Effective integration of supportability design criteria into computer

CASI Technical Report Server
CASI Technical Report Server Search Results
Wing design for light transport aircraft with imp

Figure 1: Windows from a CALVIN run: CALVIN's suggester window and context update window, with windows for a suggested web page and related concept map.

Although the system has specialized ways of dealing with resources such as web browsers, it also has generic methods for receiving and recording information sent from any application to a TCP/IP port. The system creates a new thread for every incoming resource, making it possible to send different resource entries concurrently.

Describing Contexts: A crucial question for organizing any case library is index selection—how lessons will be characterized and compared to make relevant retrievals. The current system uses three classes of indices to describe the research context for retrieving relevant resource cases: the user-specified *task* (e.g., wing design), a set of *topics* on which the user is seeking information (e.g., fuel efficiency), and an automatically-generated description of the *content* of the resource that the user consulted and considered relevant to the task and topic. Indices describing the task and target context are entered manually, using an index menu augmented with the capability to enter new projects and topics as needed. Resource content is described by a set of keywords automatically collected from resources being consulted. After filtering by a "stop list" of common words, a user-defined number of most frequent keywords is added to the context.

As the user browses documents, CALVIN maintains a global context consisting of the user-selected task and topic indices and the keywords extracted from the resources the user has viewed. The current simple similarity metric for retrieving similar contexts is based on Tversky's (1977) Contrast Model of categorization. If J and G are contexts, and their attributes are J_{Task}, J_{Topic}, etc., then their similarity is defined by

$$s(J,G) = \sum_{i \in \{Task, Topic, Content\}} P_i(\alpha|J_i \cap G_i| - \beta|J_i - G_i| - \gamma|G_i - J_i|)$$

where α, β , and γ , and P_i are constants. The system initially sets these values to default settings to simply count the number of shared attributes, but the user may modify this at runtime, and different weighting schemes may be named and stored. We intend to explore alternative and multimodal similarity assessment methods from CBR and information retrieval.

If the similarity between the two contexts exceeds a user-defined threshold, the contexts are considered similar. As the user browses, the Suggester window automatically presents a list of resources consulted in similar contexts, as shown in Figure 1. The user may select any of these to see the resource path followed to reach that resource, as shown in the box at the lower right portion of the Suggester window.

What the System Learns

CALVIN learns three types of lessons. First, the system captures lessons about the research resources that may be worth consulting in a given context: It stores cases containing information about the resources previous users found useful in similar contexts. (To avoid saving bad choices, CALVIN provides the user with the option of deleting selected cases at the end of the research session.) Second, it captures lessons that the user chooses to enter as annotations to the resources (e.g., the user might annotate the reference for a design with a warning about an unexpected problem discussed in another source). Third, CALVIN is linked to a concept mapping component for recording, elucidation, and sharing of lessons about important concepts and relationships in the domain, based on the information found in the research resources.

Concept mapping (Novak & Gowin 1984) was first applied in educational contexts, as a method for helping students to elucidate and examine their knowledge. A concept map is a two-dimensional visual representation containing nodes for concepts and named links expressing their relationships. Although such representations are initially reminiscent of semantic nets, concept maps are less constrained and need not be organized hierarchically—they simply make explicit a particular set of relationships. Part of a sample concept map is visible in the lower left corner of Figure 1. We have previously applied concept mapping to aerospace design, to support experts clarifying their own conceptualizations and to make those conceptualizations available for examination by the expert or others (e.g., members of a design team seeking to understand the expert's design to evaluate or modify it, or novices seeking to increase their own understanding). For reasons of space, we refer the reader to (Cañas, Leake, & Wilson 1999) for a description of that work. CALVIN is linked to interactive concept mapping tools developed at the University of West Florida. Concept maps can be recorded as resources to be stored, are suggested by the system as new resources, and can be accessed directly from CALVIN.

Relationship to Previous Research

The CALVIN system is related both to just-in-time case-based support systems (Leake *et al.* 1999) and to *active lessons delivery systems* (Weber *et al.* 2000), lessons learned systems which interject useful information as appropriate during the decision-making task.

It is also closely related to systems that guide web browsing based on prior browsing patterns (e.g., (Jaczynski & Trousse 1998; Lieberman 1995)). However, it differs from pure browsing systems in being designed for application to a broader set of resources and in combining an explicit task specification with the contextual information gathered from the browsing process. In addition, it not only presents information, but also supports its annotation and synthesis to provide new information resources, through the concept mapping process.

Next Steps

Planned refinements of CALVIN fall into three main categories. The first concerns context: how to refine the process for extracting contextual information (e.g., developing an indexing hierarchy, inferring task-based indices automatically and using a thesaurus to map keywords to elements of the indexing vocabulary, augmenting the context descriptions to enable direct descriptions of the tasks for which information will be used, and refining the similarity measure), and refining strategies for updating the global context in response to resources visited.

The second concerns integration: extending the scope of the system's knowledge capture and the range of its suggestions to a wide range of applications (e.g., a macro in Microsoft Office could add or update a resource case whenever a file is opened/closed/saved, or could enter resource cases for all the information resources used in generating a spreadsheet.)

The third concerns user monitoring. One goal is to move beyond simply capturing the resources the user looked at, or those the user requested, to infer whether those resources were useful and then favor presentation of useful cases (e.g., cases for resources that prompt the user to define a new concept map).

Summary

The lessons learned from problem-solving include both lessons about the domain and lessons about how to find information that is useful to the problem-solver—information about the resources that are useful in particular contexts. This paper describes ongoing research on a lessons learned system to automatically capture task-relevant resource information and proactively provide suggestions to support the research process. The approach integrates aspects of "just-in-time" task-based information retrieval, case-based reasoning, and concept mapping to guide the user towards useful resources, to learn resources to suggest in the future, and to support the elucidation and sharing of the user's new understanding.

References

Bagg, T. 1997. RECALL: Reusable experience with case-based reasoning for automating lessons learned. http://hope.gsfc.nasa.gov/-RECALL/homepg/recall.htm.

Cañas, A.; Leake, D.; and Wilson, D. 1999. Managing, mapping, and manipulating conceptual knowledge. In *Proceedings of the AAAI-99 Workshop on Exploring Synergies of Knowledge Management and Case-Based Reasoning*, 10–14. Menlo Park: AAAI Press.

Jaczynski, M., and Trousse, B. 1998. WWW assisted browsing by reusing past navigations of a group of users. In Cunningham, P.; Smyth, B.; and Keane, M., eds., *Proceedings of the Fourth European Workshop* *on Case-Based Reasoning*, 160–171. Berlin: Springer Verlag.

Leake, D.; Birnbaum, L.; Hammond, K.; Marlow, C.; and Yang, H. 1999. Integrating diverse information resources in a case-based design environment. *Engineering Applications of Artificial Intelligence* 12(6):705–716.

Lieberman, H. 1995. Letizia: An agent that assists web browsing. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence.* San Francisco, CA: Morgan Kaufmann.

Novak, J., and Gowin, D. 1984. *Learning How to Learn.* New York: Cambridge University Press.

Sary, C., and Mackey, W. 1996. Implementing RECALL: A case-based reasoning system for the access and reuse of lessons learned. In *Proceedings of the Sixth Annual International Symposium of the National Council on Systems Engineering.* St. Louis: International Council on Systems Engineering.

Tversky, A. 1977. Features of similarity. *Psychological Review* 44(4).

Watson, I. 1997. *Applying Case-Based Reasoning: Techniques for Enterprise Systems.* San Mateo, CA: Morgan Kaufmann.

Weber, R.; Aha, D.; Branting, L.; Lucas, J.; and Becerra-Fernandez, I. 2000. Active case-based reasoning for lessons delivery systems. In *Proceedings of the AAAI-2000 Workshop on Intelligent Lessons Learned.* Menlo Park: AAAI Press. In press.