An Intelligent Lessons Learned Process

Rosina Weber[T], David W. Aha[†], Hector Muñoz-Ávila[‡], & Leonard A. Breslow[†]
[T]Department of Computer Science, University of Wyoming, Laramie, WY 82071-3682
[†]Navy Center for Applied Research in Artificial Intelligence,
Naval Research Laboratory (Code 5515), Washington, DC 20375
surname@aic.nrl.navy.mil
[‡]Department of Computer Science, University of Maryland, College Park, MD 20742-3255
surname@cs.umd.edu

Abstract. A learned lesson, in the context of a pre-defined organizational process, summarizes an experience that should be used to modify that process, under the conditions for which that lesson applies. To promote lesson reuse, many organizations employ *lessons learned processes*, which define how to collect, validate, store, and disseminate lessons among their personnel, typically by using a standalone retrieval tool. However, these processes are problematic: they do not address lesson reuse effectively. We demonstrate how reuse can be facilitated through a representation that highlights reuse conditions (and other features) in the context of lessons learned systems embedded in targeted decision-making processes. We describe a case-based reasoning implementation of this concept for a decision support tool and detail an example.

1 Lessons learned process[1]

Lessons learned (LL) processes (Weber et al., 2000b) are knowledge management (KM) solutions for sharing and reusing knowledge gained through experience (i.e., *lessons*) among an organization's members. LL systems are motivated by the need to preserve an organization's knowledge and convert individual knowledge into organizational knowledge so that, when experts become unavailable; other employees who encounter conditions that closely match some lesson's context may benefit from applying it. Therefore, a lesson learned is a validated working experience that, when applied, can positively impact an organization's processes. While some organizations can quickly update the processes targeted by lessons, thus eliminating the need for a repository of lessons, other organizations (e.g., the US military, the Department of Energy) do not have this luxury (i.e., they cannot easily update their processes), which necessitates using LL systems to explicitly store and retrieve lessons.

LL systems are ubiquitous; we easily located[2] over 40 of them on the WWW, are aware that many others are used in private industry, and discovered that they rarely succeed in promoting knowledge reuse/sharing for two reasons (Weber et al., 2000b). First, the selected representations of lessons typically are not designed to facilitate reuse, either because they do not clearly identify the process to which the lesson applies, its contribution to that process, or its pre-conditions for application. Second, these systems are usually not integrated into an organization's decision-making process, which is the primary requirement for any solution to successfully contribute to KM activities (Reimer, 1998; Leake et al., 1999; Aha, 1999).

---

[1] In *Twelvth International Symposium on Methodologies for Intelligent Systems* (ISMIS'00).
[2] Our compiled findings are posted at www.aic.nrl.navy.mil/~aha/lessons.

KM solutions usually involve both organizational dynamics and technological components. We propose a *technological* solution to designing LL systems that includes a lesson representation chosen to potentiate knowledge sharing in an embedded system in which lessons are proactively brought to the attention of users. In the remainder of this paper we summarize research on LL systems, introduce a representation for lessons that promotes knowledge sharing, discuss the lessons learned process, describe the design of an *active lessons delivery* system, and detail an example of its use as a module in HICAP[3] (Muñoz-Avila et al., 1999), a decision support tool for interactive plan authoring.

2 Related work

Although dozens of lessons learned centers and their respective systems exist, few researchers have addressed LL systems, and almost none in artificial intelligence.[4] This is somewhat surprising, given that their developers and users overwhelmingly agree that current LL systems are insufficient. That is, there are several unanswered research issues regarding intelligent LL systems that need to be addressed.

Several KM publications have reported on issues related to lessons learned systems (van Heijst et al., 1996; O'Leary, 1998; Secchi, 1999; Habbel et al. 1999, SELLS, 1999). However, few of these discussed topics related to *intelligent systems* (e.g., van Heijst et al. (1997) stress the relationship between case-based reasoning (CBR) and LL systems). The only *deployed* application that uses CBR technology is NASA's RECALL system (Sary & Mackey, 1995), although three research groups have recently *proposed* CBR approaches that promote knowledge sharing.

First, the Air Campaign Planning Advisor (ACPA) (Johnson et al., 2000) disseminates videotaped stories (e.g., best practices) in a planning environment. However, ACPA does not reason on the stories, nor highlight reuse components or conditions. Thus, the user must decide whether or not to apply the memory captured in the story according to their interpretation of it.

Second, CALVIN (Leake et al., 2000) captures lessons concerning which online information resources should be searched for a given research topic. The subject and research results are used to index lessons so that when a user starts a search, previously stored results are proactively brought to the user's attention. Unlike most LL systems, CALVIN is task-specific rather than organization-specific.

Finally, we propose the Active Lessons Delivery System (ALDS), whose implementation in HICAP is discussed and exemplified in this paper. Users can interact with HICAP to author plans by iteratively decomposing complex tasks into primitive actions. ALDS monitors changes in the plan and plan state (i.e., described by a set of <question, answer> pairs), and triggers a lesson when its applicable task matches a task in the (evolving) plan and its conditions closely match the plan state. ALDS differs from the previous two embedded architectures in that (1) it focuses specifically on organizational lessons in the context of planning tasks, (2) it

[3] For more information and demonstrations of HICAP and ALDS, both developed in Java 1.2, please see http://www.aic.nrl.navy.mil/hicap.

[4] This motivated us to organize the AAAI'00 Intelligent Lessons Learned Workshop, whose homepage is www.aic.nrl.navy.mil/AAAI00-ILLS-Workshop.

automatically determines a triggered lesson's interpretation for the evolving plan, and (3) it allows users to automatically implement a lesson by pressing a button.

3 Lessons learned knowledge representation

In our survey of LL systems (Weber et al., 2000b), we found that lessons are often represented inadequately, preventing them from being easily reused or understood. For example, recorded lessons often do not highlight the task for which they apply, or precisely specify their triggering conditions. Also, free text representations, which are used in all the deployed LL systems we have found, complicate reuse because this text has to be correctly interpreted to ensure proper lesson reuse.

A lesson is derived from an experience in which the *result* derived from applying an *originating action* yields significant new knowledge (i.e., a *contribution*), due to a success or failure, that can, and should, be taught to others. A lesson's *conditions for reuse* are the relevant state variables that existed when the originating action occurred. An ideal, validated lesson facilitates its dissemination by clearly stating its contribution and the *decision*, *task*, or *process*[5] for which, by applying its *recommended response action* (i.e., a *suggestion*), a user can reduce or eliminate the potential for failures or mishaps, or reinforce a positive result. In more detail, the features of a lesson that target improvements to planning tasks are:

Originating action: The action taken in the lesson's initiating experience.

Result: This indicates whether the experience was positive or negative, and helps to determine whether to recommend repeating or avoiding the same experience.

Lesson contribution: This is the crucial feature (e.g., a set of constraints) that characterizes the originating action and is responsible for the result of the original experience. The lesson's contribution is the element that should be repeated, in conjunction with the originating action, when the experience has a positive result, and it should be avoided when the result is negative.

Applicable task: This is a pre-defined task in an organization's targeted planning process. The lesson author must identify the task to which the lesson is applicable.

Conditions for reuse: These are the values of the state that, when matched closely, will cause a lesson to be reused. Knowledge for identifying and assessing similarity between conditions and state variables must be elicited from domain experts.

[5] In decision-making systems, lessons are applicable to decisions. In planning, lessons are applicable to tasks.

Suggestion: This is the recommended response action. It is entailed by a lesson's other features (i.e., a negative experience should be avoided) and provided by the lesson author.

We illustrate this representation with a lesson from the Joint Unified Lessons Learned System[6] concerning non-combatant evacuation operations (Section 5.2). This lesson refers to the step in which non-combatants had to be registered prior to evacuation in a disaster relief operation after the April 1991 eruption of Mt. Pinatubo in the Philippines. The lesson's summary is: *The evacuee registration process was very time consuming and contributed significantly to delays in throughput and to evacuee discomfort under tropical conditions.* Our representation for this lesson is as follows:

Originating action	➔ *Evacuee registration*
Action result	➔ *Delays, time consuming, and evacuee discomfort* ➔ *negative*
Contribution	➔ *Triple registration process is problematic*
Applicable task	➔ *Evacuee registration*
Conditions	➔ *Under tropical conditions*
Suggestion	➔ *Locate an INS (Immigration and Naturalization Service) screening station at the initial evacuation processing site. Evacuees are required to clear INS procedures prior to reporting to the evacuation processing center.*

This lesson refers to a negative outcome (e.g., evacuee discomfort). The expression "under tropical conditions" is a condition for reuse. In this lesson, the applicable task is the same as the originating action, although this is not true for all lessons. The lesson recommends an alternative method of registration that is not time consuming, which defines its suggestion.

Because a lesson may still be applicable even when its *conditions* are not perfectly matched by the state, reusing lessons using a CBR approach is appropriate. The similarity assessment between conditions and state variables is modeled using elicited expert knowledge. Adaptation (e.g., replacing *tropical condition*s with *winter conditions)* is not supported because the user must decide whether to apply the recommended suggestion. The only feature that can be inferred is the *suggestion*, from information embedded in the *originating action*, *lesson contribution*, and *result*.

In the implementation of ALDS in HICAP, the applicable planning task and conditions are used for indexing a lesson.. To improve retrieval and consequently improve reuse, an effective indexing should anticipate the end users' needs and indexing style (Kolodner, 1993). Therefore, a different indexing strategy is required to facilitate retrieval of lessons that target technical decision makin*g*. This indexing strategy should use an expert's model so that technicians can identify the model component targeted by the lesson (instead of identifying an applicable task) and other features (e.g., the problem, its causes, and the symptoms associated with that component).

[6] https://www-secure.jwfc.acom.mil/protected/jcll.

4 Lessons learned process

In Section 1 we identified two problems with traditional lessons dissemination approaches: lesson representations that do not promote reuse and standalone retrieval tools. In Section 3 we proposed a representation that facilitates lesson reuse. This section focuses on embedding LL systems in their targeted processes.

An organization's lesson learned process typically involves the following tasks: *collecting*, *validating*, *storing*, *disseminating*, and *reuse*. For example, military organizations request their members, after completing a mission, to submit lessons to a *LL center*, where they are analyzed, indexed according to a task list specific to that branch of the armed services, validated, and stored in a repository. Lesson repositories are provided to military personnel, and are accessible on the secure military network SIPRNET and also on CD-ROMs. An accompanying search engine is used to submit queries in the hope of retrieving relevant lessons. Thus, LL centers are responsible for *collecting*, *validating*, *storing*, and *disseminating* lessons so potential users can *reuse* them. These five steps summarize the standard LL process, which varies slightly among LL centers.

Most systems for lesson retrieval are standalone and passive, and thus ill suited for promoting lesson dissemination and reuse because they require users to master a new process (i.e., search for relevant lessons in a separate standalone LL retrieval tool) that is independent of their problem-solving task. In fact, this process makes several unrealistic assumptions: it assumes that a user is reminded of the potential utility of a LL system whenever it may be useful, knows that the system exists, knows where to find it, has the time and the skills to use it, and can correctly interpret and reuse retrieved lessons.

We identified two desired characteristics of a LL process for facilitating kn on (e. in the is pr he cu), an ill all ly mo es em

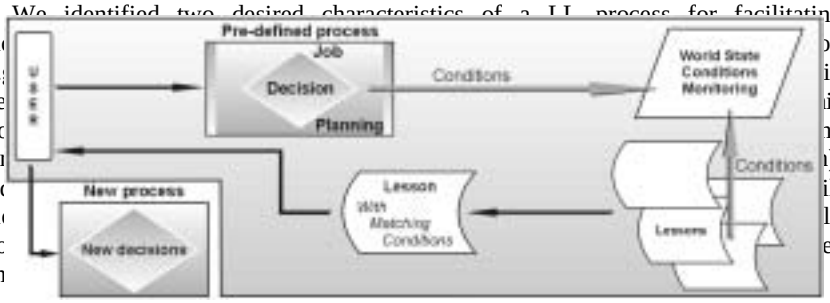

Fig. 1 Proposed lessons learned process.

These observations motivated us to design an active lessons delivery approach, to be embedded in a user's decision support tool. By automatically bringing relevant lessons to the user's attention, it promotes lesson reuse by reducing the burden on the user. In particular, this process can clarify how a lesson is relevant to the user's current decision-making task by reducing or eliminating problems of lesson interpretation and selection, does not require the user to consult a separate LL system, should increase the precision and recall of lesson retrieval, and should allow users to automatically incorporate a triggered lesson's suggestion into the evolving plan.

5 An Active Lessons Delivery System

An embedded active lessons delivery module monitors a decision-making process, bringing lessons to the user's attention when they become relevant. The primary constraint on the embedding decision support tool is that it represents and maintains information on this process that can be used to index appropriate lessons (e.g., lesson's task and triggering conditions). We illustrate this active lessons delivery approach for a military planning process in the context of HICAP. The following subsections introduce HICAP and detail an example that illustrates the use of ALDS.

5.1 The decision support tool: HICAP

HICAP (Hierarchical Interactive Case-based Architecture for Planning) (Breslow et al., 2000) helps users to formulate a hierarchical plan, which is represented as a tuple $P = \{T,R,A\}$. $T=\{T,<,\wedge\}$ is a *hierarchical task network* (HTN), where each task $t \in T$ is defined by its name t_n and duration t_d, the relation $<$ defines a (partial) temporal ordering on tasks, and $t \wedge t'$ means that t is a parent of t' in T. The leaves of T comprise the primitive actions to be included in the plan. R, which is also represented using an HTN, is the plan's set of resources. Finally, A is a set of assignments between the plan's tasks and resources. Also of interest is $S=\{<q,a>^+\}$, which denotes state information in the form of a set of <question,answer> pairs.

HICAP's modules include, among others, a *Hierarchical* Task Editor (HTE) that allows users to edit a plan, a conversational case retriever (NaCoDAE/HTN) that allows users to *interactively* select a stored decomposition to apply to a task in T, and a generative planner (JSHOP) that can be selected to *automatically* decompose tasks in T into subtasks. The plan state S is updated by direct user input, through user interactions with NaCoDAE/HTN, or by JSHOP.

5.2 The task domain: Noncombatant evacuation operations (NEOs)

We initially designed HICAP for deliberative NEO planning; no AI system has been deployed to assist military experts to plan NEOs. NEOs (DoD, 1994) are performed by the US military to assist in the evacuation of non-combatants, non-essential military personnel, and others (e.g., host nation citizens) whose lives are in danger (e.g., due to political insurgencies, volcanic eruptions) from an endangered location (e.g., a beleaguered US embassy) to an appropriate safe haven.

Each lesson in HICAP is indexed by its *applicable task* and *conditions*. For example, one such lesson for the NEO planning domain is:

Originating action ➔ *Assign conventional use of air wing*
Action result ➔ *Increases the risk to detection of clandestine SOF* ➔*negative*
Contribution ➔ *Conventional (low visibility) air wing increases SOF risk*
Applicable task ➔ *Assign air wing*
Conditions ➔ *Q: Is it necessary to use covert SOF helicopters? A: Yes*
Suggestion ➔ *Assign high visibility to conventional air wing*

A lesson's *conditions* are represented as <question,answer> pairs so their similarity with state variables can be easily assessed. The user decides whether and how the lesson's suggestion will be implemented, as illustrated below.

5.3 Active lessons delivery module: An example

For this example, we use the fictitious *Terror in the Jungle* NEO scenario, obtained from the DISA *Adaptive Courses of Action* ACTD.[7] Some tasks in its task hierarchy can be further decomposed using interactive case retrieval. After the user selects a task to expand, NaCoDAE/HTN displays alternative expansions that could apply, along with questions that, if answered by the user, could help determine which case's conditions best matches *S*. The task being expanded here is *Rescue mission*, which concerns how to safely evacuate the evacuees.

After answering some questions and thus updating the state, the case retriever then displays the question *Is it necessary to use covert SOF helicopters?* The user answers *Yes*, yielding a perfect match with a task decomposition case that expands to the subtasks *Use ground support* and *Assign conventional use of air* wing.

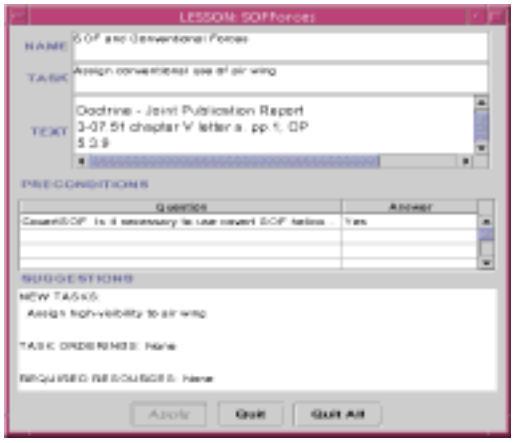

Fig. 2. A lesson pertaining to camouflaging special operations forces.

[7] http://www.les.disa.mil/insert/acoa/index.htm

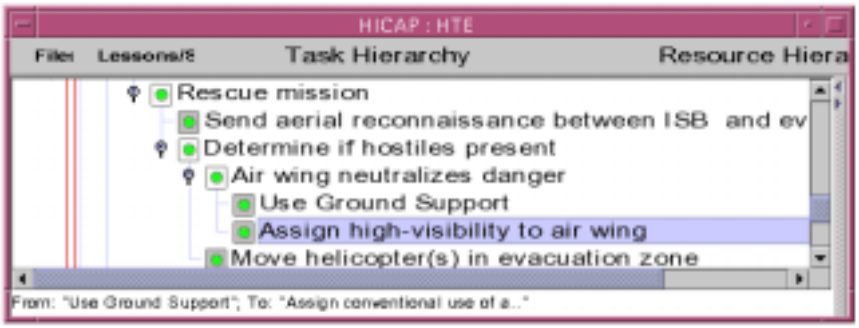

Fig. 3. A subset of the task hierarchy after applying the lesson shown in Figure 2.

6 Concluding Remarks and Future Work

In this paper we focused on the reuse of lessons learned. We identified two problems that interfere with lesson reuse: inadequate lesson representations (e.g., how different features should be highlighted to enable interpretation) and system architecture (i.e., how lessons learned systems should be embedded into the decision-making process). We then proposed an active lessons delivery approach (ALDS) to address these problems and exemplified its use in HICAP, a plan authoring tool.

We have not yet evaluated the utility of ALDS in NEO exercises, and instead developed a simple travel planning domain for evaluating the impact of ALDS (Weber et al., 2000a). In future work, we will examine how to use HICAP to guide interactive lesson elicitation, demonstrate the utility of active lessons delivery for other decision support tasks, and transition HICAP to the ACOA ACTD project.

Acknowledgements

This research was supported by grants from the Office of Naval Research, the Naval Research Laboratory, and the University of Wyoming.

References

1. Aha, D.W. (1999). The AAAI-99 KM/CBR workshop: Summary of contributions. In C. Gresse von Wagenheim & C. Tautz (Eds.) *Proceedings of*

the ICCBR-99 Workshop on Practical Case-Based Reasoning Strategies for Building and Maintaining Corporate Memories. Munich: Unpublished.

2. Breslow, L.A., Muñoz-Avila, H., Aha, D.W., Weber, R., & Nau, D. (2000). *HICAP: Hierarchical interactive case-based architecture for planning* (Technical Report AIC-00-006). Washington, DC: NRL, NCARAI.
3. DoD (1994). *Joint tactics, techniques and procedures for noncombatant evacuation operations* (Joint Report 3-07.51). Washington, DC: Department of Defense.
4. Habbel, R., Harter, G., & Stech, M. (1999). Knowledge management: knowledge-critical capital of modern organizations. *Booz Allen & Hamilton Insights*. [www.bah.com/viewpoints/insights/cmt_knowmanage_2.html]
5. Johnson, C., Birnbaum, L., Bareiss, R., & Hinrichs, T. (2000). War Stories: Harnessing Organizational Memories to Support Task Performance. *Intelligence*: *New Visions of AI in Practice*, *11*(1), 17-31.
6. Kolodner, J. (1993). *Case-based reasoning.* San Mateo, CA: Morgan Kaufmann.
7. Leake, D.B., Bauer, T., Maguitman, A., & Wilson, D.C. (2000). Capture, storage, and reuse of lessons about information resources: Supporting task-based information search. In D.W. Aha & R. Weber (Eds.) *Intelligent Lessons Learned Systems: Proceedings of the AAAI Workshop* (Technical Report WS-00-08). Menlo Park, CA: AAAI Press.
8. Leake, D., Birnbaum, L, Hammond, K., Marlow, C., & Yang, H. (1999). Task-based knowledge management. In D.W. Aha, I. Becerra-Fernandez, F. Maurer, & H. Muñoz-Avila (Eds.) *Exploring Synergies of Knowledge Management and Case-Based Reasoning: Proceedings of the AAAI-99 Workshop* (Technical Report WS-99-10). Orlando, FL: AAAI Press.
9. Muñoz-Avila, H., McFarlane, D., Aha, D.W., Ballas, J., Breslow, L.A., & Nau, D. (1999). Using guidelines to constrain interactive case-based HTN planning. *Proceedings of the Third International Conference on Case-Based Reasoning* (pp. 288-302). Munich: Springer.
10. O'Leary, D.E. (1998). Enterprise Knowledge Management. *Computer*, *31*(3), 54-61.
11. Reimer, U. (1998). Knowledge integration for building organisational memories. *Proceedings of the Eleventh Banff Workshop on Knowledge Acquisition.*. [http://ksi.cpsc.ucalgary.ca/KAW/KAW98/KAW98Proc.html]
12. Sampson, M. (1999). NASA parts advisories – nine years of experience, and counting. In (Secchi, 1999).
13. Sary, C., & Mackey, W. (1995). A case-based reasoning approach for the access and reuse of lessons learned. *Proceedings of the Fifth Annual International Symposium of the National Council on Systems Engineering* (pp. 249-256). St. Louis, Missouri: NCOSE.
14. Secchi, P. (Ed.) (1999). *Proceedings of Alerts and Lessons Learned: An Effective way to prevent failures and problems* (Technical Report WPP-167). Noordwijk, The Netherlands: ESTEC.
15. SELLS (1999). *Proceedings of the Society for Effective Lessons Learned Sharing Spring Meeting*. Las Vegas, NV: Unpublished. [www.tis.eh.doe.gov/ll/sells]
16. van Heijst, G., Hofman, M., Kruizinga, E., & van der Spek, R. (1997). AI-techniques and the knowledge pump. In B. Gaines & R. Uthursamy (Eds.)

Artificial Intelligence in Knowledge Management: Proceedings of the 1997 Spring Symposium (Technical Report SS-97-01). Menlo Park, CA: AAAI Press.

17. van Heijst, G., van der Spek, R., & Kruizinga, E. (1996). Organizing corporate memories. *Proceedings of the Tenth Banff Workshop on Knowledge Acquisition.* Banff, Canada. [ksi.cpsc.ucalgary.ca/KAW/KAW96/KAW96Proc.html]
18. Weber, R., Aha, D.W., Muñoz-Avila, H., & Breslow, L.A. (2000a). Active delivery for lessons learned systems. To appear in *Proceedings of the Fifth European Workshop on Case-Based Reasoning*. Trento, Italy: Springer.
19. Weber, R., Aha, D.W., & Becerra-Fernandez, I. (2000b). *Intelligent lessons learned systems.* To appear in *International Journal of Expert Systems Research & Applications.*