Computational Intelligence, Volume 17, Number 2, 2001

MAINTAINING CASE-BASED REASONERS: DIMENSIONS AND DIRECTIONS

DAVID C. WILSON AND DAVID B. LEAKE
Computer Science Department, Indiana University

Experience with the growing number of large-scale and long-term case-based reasoning (CBR) applications has led to increasing recognition of the importance of maintaining existing CBR systems. Recent research has focused on case-base maintenance (CBM), addressing such issues as maintaining consistency, preserving competence, and controlling case-base growth. A set of dimensions for case-base maintenance, proposed by Leake and Wilson, provides a framework for understanding and expanding CBM research. However, it also has been recognized that other knowledge containers can be equally important maintenance targets. Multiple researchers have addressed pieces of this more general maintenance problem, considering such issues as how to refine similarity criteria and adaptation knowledge. As with case-base maintenance, a framework of dimensions for characterizing more general maintenance activity, within and across knowledge containers, is desirable to unify and understand the state of the art, as well as to suggest new avenues of exploration by identifying points along the dimensions that have not yet been studied. This article presents such a framework by (1) refining and updating the earlier framework of dimensions for case-base maintenance, (2) applying the refined dimensions to the entire range of knowledge containers, and (3) extending the theory to include coordinated cross-container maintenance. The result is a framework for understanding the general problem of case-based reasoner maintenance (CBRM). Taking the new framework as a starting point, the article explores key issues for future CBRM research.

Key words: case-based reasoning; case-base maintenance; case-based reasoner maintenance; knowledge containers; metamaintenance.

1. INTRODUCTION

The growing use of large-scale and long-term case-based reasoning (CBR) applications has brought with it increased awareness of the importance of maintaining CBR systems. Large-scale CBR systems have become more prevalent, with case library sizes ranging from thousands (e.g., Cheetham 1997, Kitano and Shimazu 1996) to millions of cases (Deangdej et al. 1996). The use of large case bases raises concerns about the utility problem for case retrieval (Francis and Ram 1993; Smyth and Cunningham 1996), in which the growing cost of case retrieval outweighs the efficiency benefits from additional cases and has prompted research on controlling case-base growth through compaction policies (Smyth and Keane 1995; Smyth and McKenna 1999a; Zhu and Yang 1999). Even for smaller case bases, the difficulties of distributed case collection (Borron et al. 1996) and use (Doyle and Cunningham 1999; Watson and Gardingen 1999) and the vagaries of real-world data raise concerns about the consistency and accuracy of case knowledge, motivating efforts to maintain the case base to improve its quality (Racine and Yang 1997). These concerns have led to active research in the area of case-base maintenance (CBM).

Many maintenance issues, however, extend beyond the case base (Leake and Wilson 1998; Heister and Wilke 1998). CBR systems depend on the knowledge contained in multiple *knowledge containers* (Richter 1998), such as similarity knowledge and adaptation knowledge, in addition to the case base. A number of projects have illuminated maintenance issues for knowledge containers outside the case base for particular systems and tasks, but there is currently no common framework to guide a more general study of maintenance in CBR.

Address correspondence to David Wilson at Department of Computer Science, University College Dublin, Belfield, Dublin 4, Ireland.

Leake and Wilson (1998) provide a framework for characterizing case base maintenance systems. This framework is aimed at understanding the state of the art in CBM, illuminating current practice and facilitating the comparison of particular approaches. In a similar line to characterizations that have proven useful for studying case adaptation (Hanney et al. 1995; Voß 1996), this framework has helped to identify problems and opportunities for study, suggesting points of exploration in the space of possible CBM systems. However, this work has broader applicability. The goals addressed by the CBM framework are common across all the CBR knowledge containers, and the dimensions defined for CBM are flexible enough that they can be extended and applied to characterize generalized maintenance across all knowledge containers in CBR.

This article extends and generalizes the CBM framework, presenting an overall perspective on maintaining CBR systems. It takes CBM as a starting point, adapting and generalizing the analysis of CBM to extend to other knowledge containers. Sections 2 and 4 refine our earlier framework of dimensions for CBM and use the framework to provide an updated description of current approaches in CBM. Section 5 generalizes the framework to apply across knowledge containers to characterize possible approaches to maintenance of all CBR system knowledge containers, in what we refer to as *case-based reasoner maintenance* (CBRM). CBRM processes can include not only the revision of knowledge in individual knowledge containers but also the coordinated updating of multiple knowledge containers, the strategic transfer of knowledge between knowledge containers, and metamaintenance—maintenance of maintenance knowledge—in order to achieve indirect revisions of knowledge containers such as lazy case-base updating. Section 8 explores the ramifications of the previous analysis and develops a vision for future CBRM research. Altogether, the article provides a framework for understanding the state of the art in CBRM.

2. DEFINING CASE-BASE MAINTENANCE

As a basis for our discussion of CBRM, we first describe and refine our framework for characterizing case-base maintenance. We define *case-base maintenance* (CBM) as the process of refining a CBR system's case base to improve the system's performance:

> Case-base maintenance implements policies for revising the organization or contents (representation, domain content, accounting information, or implementation) of the case-base in order to facilitate future reasoning for a particular set of performance objectives [Leake and Wilson 1998].

Note that this definition considers the information defining an indexing scheme to be an intrinsic organizational component of the case base itself. Thus CBM may involve revising indexing information, links between cases, or other organizational structures and their implementations.

Maintaining case-base contents may affect a single case or multiple cases. It may revise the case representations used (e.g., changing the predicates used to describe domain features), may revise either domain information in the case base (e.g., correcting an erroneous feature in a case or adding or deleting an entire case) or "accounting" information (e.g., changing information about how frequently a case has been accessed), or may revise how case representations are implemented (e.g., changing from lists to feature-vectors). Thus maintenance of case-base contents may revise the case base at the implementation level, representation level, or the knowledge level (see Dietterich 1986).

3. CBM PERFORMANCE OBJECTIVES AND CONSTRAINTS

The performance objectives for a CBR system provide criteria for evaluating the internal behavior and task performance of a particular system for a given initial case base and sequence of problems solved. The choice of CBM strategies is driven by the maintainer's performance goals for the system and by constraints on the system's design and the task environment. In general, there will be multiple performance measures for a CBR system, and there is no guarantee that all of them can be maximized simultaneously. Smyth and McKenna (1999) define three types of top-level goals for CBR systems:

1. Problem-solving efficiency goals (e.g., average problem-solving time)
2. Competence goals (the range of target problems solved)
3. Solution quality goals (e.g., the error level in solutions)

These goals may give rise to quantitative maintenance goals (e.g., achieving particular problem-solving time or limits on case-base size) or qualitative ones (e.g., to extend system competence). Smyth (1998) provides compelling arguments for the importance of shaping maintenance policies according to a complete set of performance objectives. Of course, performance objectives may change over time to reflect varying external circumstances, which may necessitate changing (maintaining) maintenance policies as well.

The application of maintenance policies to achieve these goals is also shaped by constraints from the external environment (Leake and Wilson 2000):

1. Case-base size limits (if any)
2. Acceptable long-term/short-term performance tradeoffs
3. The expected distribution of future problems
4. The availability of secondary sources of cases

For example, Smyth and Keane's (1995) competence-preserving deletion strategies reflect all these constraints. Their deletion process keeps the case base within acceptable size limits (constraint 1), their competence-guided choices are intended to minimize the loss of future coverage (constraint 2), their methods' deletion choices assume a uniform distribution of problems (constraint 3), and no other sources of cases are available for recovering deleted information (constraint 4), making preservation of competence a key concern. Other instantiations of these constraints would give rise to different strategies. For example, if short-term performance is crucial, long-term performance is less important, and current problems are concentrated in a small part of the case base, it may be acceptable to sacrifice current competence and build it back through future learning. Thus a case-based reasoner needs policies for achieving its maintenance goals in light of its constraints. The following section develops a characterization of the properties of CBM policies.

4. A FRAMEWORK FOR DESCRIBING CBM POLICIES

The goal of a categorization scheme for CBM is threefold. First, by identifying classes of similar maintenance approaches, such a categorization scheme can shed light on the state of current practice in the field, increasing understanding of current CBM approaches. Second, mapping out the space of candidate approaches helps identify parts of the space that have not been addressed in previous work; these gaps, in turn, suggest research opportunities. Third, a categorization scheme for maintenance approaches is a first step toward cataloging the most appropriate approaches for particular performance goals.

Our framework categorizes CBM approaches in terms of *case-base maintenance policies* that determine when and how a CBR system performs CBM. Maintenance policies are described in terms of how they gather data relevant to maintenance, how they decide when to trigger maintenance, whether they react to problems or proactively forestall them, the types of maintenance operations available, and how selected maintenance operations are executed.

In the framework, *data collection* gathers, synthesizes, and distills the data about the case base and about system processing; this is the information that will be used to determine whether maintenance operations should be performed. *Triggering* takes this information as input, makes the decision whether maintenance is needed, and selects maintenance actions from a range of possible *operation types*. *Execution* describes when and how the selected revisions are actually applied to the case base.

Descriptions generated using the framework characterize basic combinations of policy attributes. A single CBR system may include multiple maintenance policies, each one implementing a different part of the system's overall maintenance agenda (e.g., Minor and Hanft 2000). The following dimensions would be used to describe each policy separately. Coordination of maintenance policies is described in Section 6.

4.1. Data Collection

Data collection gathers information about individual cases, about the case base in part or as a whole, and/or about the overall processing behavior of the CBR system. Data collection about individual cases might record the number of times a case has been used successfully or the number of times it has failed. Data collection about the case base as a whole could involve, for example, monitoring the size of the case base. Data collection about processing might involve noting clusters in input problems, input problems that the system is unable to solve successfully, or input problems for which processing costs are too high.

Type of Data: None, Synchronic, or Diachronic. There are three approaches to collecting and analyzing data to decide when CBM is needed. The simplest is to do no collection at all. A policy with no data collection makes maintenance decisions independently of the present or past state of the case base. As such, this type of policy is referred to as *nonintrospective*. For example, a CBR system that updates its case base by unconditionally adding a case each time it adapts a prior case would need no data collection. This is the approach of most CBR systems. Similarly, a system may drive maintenance according to external information sources. This is valuable for proactive maintenance, for example, to add cases to a help-desk case base in anticipation of future queries.

More sophisticated reasoning is enabled by considering a snapshot of the current case base in part or as a whole. Examination of this information can determine, for example, whether a case is worth adding to a case base because it increases the competence of the CBR system or whether a solution can be discarded without affecting competence (Smyth and Keane 1995). As another example, Reinartz et al. (2000) propose a set of measures that can be computed to assess the overall quality of a case base in order to trigger maintenance. Policies that consider snapshot information are called *synchronic*.

The most informative approach is to collect data over time, over a sequence of snapshots, in order to identify trends in how case-base contents and usage are changing. Policies that consider changes in the case base over time are called *diachronic*. For example, a policy that gathered information about trends in retrieval times to identify the onset of utility problems would be diachronic. Because synchronic and diachronic collection examine the internal state of the case base, both are referred to as *introspective*.

Timing: Periodic, Conditional, or Ad Hoc. A maintenance policy must specify when data collection is performed. In our framework, there are three possibilities. *Periodic* timing happens at a set frequency with respect to the CBR cycle. For example, data collection might be performed after each problem-solving cycle. Periodic timing that happens every cycle is termed *continuous*. *Conditional* data collection is performed in response to a well-defined but nonperiodic condition. For example, analysis might be triggered whenever the number of cases in the case library reaches a particular threshold (Smyth and Keane 1995). *Ad hoc* timing happens under ill-defined conditions determined externally to the CBR system.[1] Examples of ad hoc timing are user-initiated tests on the case base to determine whether maintenance is needed or a domain expert's decision to add new cases regardless of the case base contents.

Integration: On-line or Off-line. Data collection may operate *on-line*, during the course of an active reasoning episode, or *off-line*, during a pause in reasoning, such as waiting for user input or when idle between reasoning episodes. The choice between on-line and off-line processing may affect the resources that can be devoted to the analysis process, making it important for determining whether a policy is appropriate for time-constrained processing.

4.2. Triggering

The results of data analysis serve as input for determining whether CBM is necessary. Both the *timing* and *integration* dimensions discussed previously apply to this step as well. Maintenance triggering evaluates whether to perform maintenance, selects maintenance actions to use, and may set parameters to guide their future execution (e.g., determining when they will be performed). Triggering can be done periodically, conditionally, or on an ad hoc basis and on-line or off-line.

Conditional triggering can be subdivided into three classes depending on the conditions that determine whether maintenance is triggered: *space-based* (e.g., filling a limited amount of case storage), *time-based* (e.g., retrieval time exceeding a threshold), or *result-based* (e.g., the system failing to solve a given problem or the wrong case being retrieved).

4.3. Proactive versus Reactive Maintenance

CBM is often seen as a process of detecting problems and responding to repair them (e.g., for case inconsistencies or exceeding case-base size limits). In this case, maintenance is triggered by conditions typically indicative of system failures. However, maintenance also may be proactive, taking steps despite successful performance, to avoid predicted future problems or to improve future performance. For example, a

[1]This category name in no way implies that the choice is ill-considered; simply that it is not under control of the policy.

software company with a case-based help-desk system might do maintenance in advance of the launch of a new product in order to seed the case base with cases expected to be useful after the product has been released.

4.4. Operation Types

Different maintenance policies revise different types of information (the *target type*) at different levels (the *revision level*).

Target Type. For CBM, revision operations can focus on three types of targets: *indexing structures*, *domain contents*, and *accounting information*.

Revision Level. Revision operations can make revisions with ramifications at three levels: affecting only the *implementation level* (e.g., changing an indexing structure from a list to a D-tree when the case base exceeds a certain size or changing case representations from lists to vectors), affecting the *representation level* (e.g., reconciling inconsistent feature names or case formats in cases that come from different sources), or affecting the *knowledge level* as well (e.g., correcting an erroneous feature value, generalizing case values, or adding or deleting cases).

Finer-grained characterizations of operator types are of course possible [e.g., Heister and Wilke (1998) describe a set of atomic maintenance operations]. However, as with the rest of the categorization scheme, we have used higher-level categories to facilitate cross-system comparisons of major characteristics.

Scope of Maintenance: Broad or Narrow. A given operation may be applied locally, to few items in the case base, or more globally. Operations that affect a single case or a small subset of the case base have *narrow* scope, and operations that affect a large subset or the entirety of the case base have *broad* scope. This dimension is especially useful when characterizing resource-bounded processing.

4.5. Execution

Execution is characterized by the timing of maintenance operations and their integration with other system processing. Execution timing is described using the timing dimension previously described for data collection (periodic, conditional, or ad hoc); timing also may be "none" for systems with no execution. For example, a maintenance policy simply may inform a maintainer that maintenance is needed without making changes (none), changes may be made on a regular basis (periodic), changes may be held for batch updating when enough cases are accumulated (conditional), or changes may be held for when an expert is available (ad hoc). Likewise, execution integration is described as on-line or off-line depending on whether maintenance operations are performed during or between reasoning episodes.

4.6. Categorizing Policies for CBM

To illustrate the use of the framework and to understand the range of CBM methods, we apply the framework to a sampling of CBM approaches, beginning with a few simple examples. In describing particular maintenance policies, we emphasize two parts of the CBM framework that we consider particularly useful for describing current CBM systems: the type of data collected and how maintenance policies are executed. Table 1 summarizes the described approaches along these dimensions.

TABLE 1. Sample CBM Approaches Placed Along Major Dimensions

			Data Collection		
			Type of Data		
Activation Timing	*Integration Time*	*Scope of Changes*	*None*	*Synchronic*	*Diachronic*
Periodic	On-line	Broad			
		Narrow	CBR_1	Muñoz-Avila	
	Off-line	Broad			
		Narrow			
Conditional	On-line	Broad		Fox & Leake	
		Narrow	Leake & $Wilson_2$	Smyth & $Keane_2$; Surma & Tyburcy; Hammond; Ihrig & Kambhampati	Leake & $Wilson_1$
	Off-line	Broad		Smyth & $Keane_1$; Portinale et al.	
		Narrow			
Ad hoc	On-line	Broad			
		Narrow		Minor & $Haft_1$	
	Off-line	Broad	CBR_2	Aha & Breslow; IBL_n; Watson; Racine & $Yang_{1,2,3}$; Netten; Smyth & $McKenna_{1,2}$; Zhu & Yang; Göker & Roth-Berghofer; Yang & Wu	
		Narrow	CBR_2	Watson	
	No Execution			Shimazu & Takishima	
			Non-introspective	*Introspective*	

Policies Targeting Domain Content. Policies targeting domain content may be divided into policies aimed at adding and deleting cases and policies aimed at revising internal case content. We first consider addition and deletion policies and then policies to refine the cases themselves.

Standard case learning and manual maintenance. The standard learning of CBR problem-solving systems (always adding each new case to the case base) is designated in Table 1 as CBR_1. No data analysis is performed—the new case is recorded without considering the existing contents of the case base—so it is nonintrospective. Because learning happens during each reasoning cycle, this policy is continuous (periodic) and on-line. Because only a single case is added, the scope of change is narrow.

Another common CBR method (CBR_2) involves a non-learning system maintained by a domain expert who sometimes adds a variable number of new cases. For this method, we presume no system analysis of the existing case base, so the maintenance policy is nonintrospective. Because the timing of the updates depends on the expert's external decision, the timing is ad hoc. Because the cases are added manually outside of normal processing, the integration is off-line. Because the number of cases can be small or large, the scope varies from narrow to broad.

Additional policies aimed at case retention. Smyth and Keane (1995) describe a competence-preserving approach to case deletion, which specifies a case utility hierarchy in terms of coverage and reachability. When the number of cases in the case base exceeds the "swamping limit," their "footprint-utility deletion" strategy selects candidates for deletion based on the utility hierarchy. Because the hierarchy is defined with respect to the current state of the case base, the policy is synchronic. Because maintenance is triggered in response to the current size of the case base, timing is conditional. Smyth and Keane describe this mechanism as being applied either to small numbers of cases during processing, using a heuristic method of utility evaluation (Smyth & Keane$_2$, on-line and narrow) or to large numbers of cases with full analysis outside the reasoning cycle (Smyth & Keane$_1$, off-line and broad).

Surma and Tyburcy (1998) describe policies for replacing older cases as new cases are learned in order to bound case-base size to maintain bounded retrieval time. The policies act on the current state of the case base (synchronic) during the storage phase (on-line) when the size limit is reached (conditional) to make a narrow change.

Smyth and McKenna (1999) present a policy for case-base editing/compaction (Smyth & McKenna$_1$) that uses an explicit case competence model based on notions of coverage and reachability. Their "relative coverage" metric provides a precise measure of competence contributions for individual cases. This allows the case set to be ordered by likely competence contribution. To build the case base, the ordered set is presented to a condensed nearest-neighbor algorithm that successively retains only those cases which are not solved by a case that has already been retained. This method examines the current state of the case base (synchronic). It is presented as a way to edit the entire case base during construction (ad hoc, off-line, broad), although they also have developed efficient methods for keeping the reachability and coverage measures current as the system is used (Smyth and McKenna 2000).

Muñoz-Avila (1999) presents a case retention policy based on retrieval benefits to case-based planning. After a problem-solving episode, adaptation effort is analyzed to determine whether the guidance of retrieved cases was "beneficial" (the new case need not be stored) or "detrimental" (the new case is added). This policy is synchronic, periodic, on-line (prestorage), and narrow.

Portinale, Torasso, and Tavano (1999) present a strategy for managing case memory by removing "useless" cases (that have not been retrieved before an expiration limit) and "false positive" cases (that have been retrieved and have had more adaptation failures than successes). Memory management is conditionally triggered after a variable-length time window that is tailored to the growth of case memory and reasoning failure rate. This policy uses synchronic information, conditional timing, and off-line integration to make broad changes.

Zhu and Yang (1999) describe a case-addition algorithm for case-base compaction that uses a problem-neighborhood model of case coverage. Cases are successively added based on added benefit/usefulness to the neighborhood of the case set retained so far. The analysis is synchronic, with ad hoc timing, off-line integration, and broad scope.

Policies aimed at internal case content. A number of proposed CBM policies are aimed at internal case content. Shimazu and Takashima (1996) describe a version of the CARET system that identifies discontinuities in a case base. This system uses synchronic data collection; it retrieves a set of "maybe similar cases" (MSCs), chooses a single best "base case" (BC), and classifies as "discontinuous" any remaining MSCs whose suggestions differ from the BC by more than a given threshold, identifying them as potential candidates for maintenance. However, the system does not execute revisions, so the policy has no execution.

Racine and Yang (1997) describe policies for identifying redundant cases (Racine & Yang_1) and inconsistent cases (Racine & Yang_2). Both policies rely on an analysis of the current state of the case base, so they are synchronic. Both are applied to the case base as a whole when desired by a case-base maintainer, so they are broad, ad hoc, and off-line.

Minor and Hanft (2000) describe a framework to support interactive revision of case content over case "life cycles" (Minor & Hanft_1). Support for revising case content is based on the current state of the case base and is interactive (ad hoc and on-line), with narrow changes being made to individual cases.

Leake and Wilson (1998) describe a maintenance policy that updates case contents with a revision policy installed in response to trends in performance anomalies (Leake & Wilson_2), enabling a lazy update of the case base. The installed policy always checks (no analysis) whether a retrieved case (on-line between retrieval and application) has been updated to reflect a previously detected trend (conditional timing) and updates just that case (narrow scope) in situ before passing it on for further reasoning.

Additional approaches address both the presence of cases in the case base and their internal content. Watson (1997) presents a set of guidelines for human case-base maintainers that involve performing periodic tests on the entire case base. This policy can be described as having synchronic analysis, ad-hoc timing, off-line execution, and narrow or broad scope.

Netten (1999) presents a framework for verification of case-base integrity in case-based diagnosis systems. This includes checks for redundancy, inconsistency, and incompleteness in case definitions, as well as verification of coverage, reachability, and accuracy. The framework is presented as a means to validate a diagnosis case base before being applied in critical environments. It makes use of the current state of the case base and is ad hoc, off-line, and broad in scope.

Göker and Roth-Berghofer (1999) describe the "maintenance cycle" of the HOMER case-based help-desk support system, which includes a policy for verifying whether new cases entered by help-desk operators should be added to the central case repository. Redundancy and inconsistency are checked by a case-base administrator. Potential cases are checked against the current case base (synchronic), at a time determined by the maintainer (ad hoc), separate from ongoing processing (off-line), and only for cases under consideration (narrow).

Policies Targeting Indices. A number of classification systems using IBL and related techniques (IBL_n) include policies for eliminating noisy and redundant instances from a set of training examples (cases). These systems generalize a case base either explicitly, by merging cases with similar coverage (e.g., Domingos 1995), or implicitly, by choosing a smaller, representative subset of cases (e.g., Aha, Kibler, and Albert 1991). Such policies typically consider a static set of cases (synchronic), are user-initiated (ad hoc), perform execution off-line, and are applied to the entire training set (broad). Because case features (other than the category) are only used as indices, we view their generalizations as revising indexing information. When IBL systems remove noisy instances or remove a class entirely, their target is domain content.

Many methods have been proposed for selecting case indices. Some are included in the standard case-addition process (CBR_1), as in the model-based approach of Bhatta and Goel (1995). Others, however, adjust current indices in response to performance deficiencies. Hammond (1989) describes a failure-driven method for explanation-based selection of new indexing features. Likewise, Ihrig and Kambhampati (1997) describe a policy that explains plan replay failures in order to add features to check during future retrievals. These policies are conditional, on-line, and make narrow changes.

Fox and Leake (1995) describe a policy that triggers index revision for plan cases in response to plan failures. This policy considers snapshot information about execution (synchronic), is executed conditionally, is performed on-line, and revises indices in the entire case base (broad scope).

Aha and Breslow (1997) describe an index revision method for conversational CBR that considers an entire case base to optimize interactive question paths in response to an external request. This policy has synchronic data collection, ad hoc activation timing, off-line integration, and broad scope.

Racine and Yang (1997) describe a policy for deriving and updating indices of unstructured cases (Racine & $Yang_3$) using methods derived from information retrieval. Like their other policies, this policy is synchronic, broad, and off-line and has ad hoc execution.

Smyth and McKenna (1999) present a method for index refinement (Smyth & $McKenna_2$) based on competence groups defined with reference to measures of coverage and reachability. From each competence group, a set of footprint cases that cover the remainder of the group is used to focus retrievals. The indexing organization uses synchronic information, ad hoc and off-line, to make broad changes.

Yang and Wu (2000) describe a method in which a large original case base is partitioned into a distributed set of case-base clusters using density-based clustering methods. Retrievals are made from the distributed case organization by finding the best case cluster and then the best case within that cluster. The clustering method uses a broad snapshot of the original case base, and the smaller case bases are described as being built by an expert based on the clustering result, ad hoc and off-line.

Policies targeting maintenance policies. Leake and Wilson (1998) describe a diachronic maintenance policy (Leake & $Wilson_1$) that detects potentially important trends in performance anomalies on-line, based on the conditional strength of the trend, and responds by installing a new maintenance policy tailored in response to the trend detected. It performs a narrow change—adding a new maintenance policy. This type of policy is described in more detail in Section 7.

5. A GENERAL FRAMEWORK FOR CASE-BASED REASONER MAINTENANCE

The CBM framework developed in previous sections is flexible enough to provide a general set of dimensions for describing maintenance systems beyond CBM alone. In developing the overall maintenance framework, we first define the more general *case-based reasoner maintenance*: Case-based reasoner maintenance (CBRM) implements policies for revising one or more knowledge containers in order to facilitate future reasoning for a particular set of performance objectives.

The same CBR performance goals described in Section 3 guide this more general process, but under more general constraints:

1. Knowledge container and component processing constraints (e.g., case-base size limits, adaptation-effort thresholds)
2. Acceptable long-term/short-term performance tradeoffs
3. The availability of secondary sources of knowledge
4. The expected distribution of future problems

Regardless of the knowledge container(s) involved, maintenance policies are determined by methods for data collection, triggering, operation types, and execution. In this section we extend the framework developed for CBM to generalized CBRM. As with CBM, the goals of this extended framework are to illuminate current practice by identifying classes of maintenance methods, to identify research opportunities where parts of the space of possibilities have not been addressed in previous work, and to help identify which maintenance approaches are most appropriate for particular performance goals. We do not claim that we provide a final taxonomy or a complete summary of CBRM but that the framework provides a useful way to describe central aspects of current practice in CBRM and identifies opportunities for future maintenance research.

5.1. Extending the CBM Framework to CBRM

Adapting the CBM framework to other knowledge containers maintains the overarching structure but requires adjustments of some specific features. The finer-grained distinctions in operation types and triggering will change according to the knowledge container being maintained, but both dimensions still apply. For example, operations for similarity maintenance may have different targets (e.g., weighting schemes), but the notion of revision level (implementation, representation, or knowledge level) still applies. One set of possible similarity operations is presented in Heister and Wilke (1998).

Conditional triggering also will have different general conditions. Triggering may be based on conditions when using a particular knowledge container (e.g., the average efficiency or quality of adaptation or retrieval) or on the results of overall reasoning. Thus the space-based CBM type of triggering may not apply for adaptation, but considerations for adaptation efficiency thresholds would. We discuss issues the selection of which knowledge container to maintain in order to meet performance goals in Section 6.

While the notion of broad versus narrow scope still applies, the implications are relative to the particular knowledge container. In the large, revising the entire case base may take far longer than revising the entire similarity weighting scheme, but both might be considered to have broad scope within their knowledge container.

5.2. Categorizing Policies for Other Knowledge Containers

Having extended the framework for CBRM, we present some examples of how the generalized framework may be applied to knowledge containers other than the case base. Here we discuss policies that affect the other three well-known knowledge containers identified in Richter (1998): similarity, adaptation, and vocabulary. We propose, however, that the same general framework could be applied to other knowledge containers—maintenance knowledge itself, for example.

Similarity Maintenance. There is a large body of work dealing with methods for learning feature weighting schemes in k-nearest neighbor classifier lazy-learning algorithms. Such policies typically consider a static set of training cases/instances (synchronic) in learning that are user-initiated (ad hoc), perform execution off-line (training happens off-line from system use), and are applied to the entire training set (broad). For a discussion and comparison of these types of methods, see Wettschereck et al. (1997).

Muñoz-Avila and Huellen (1996) describe a policy that analyzes adaptation effort after each problem-solving episode in order to adjust feature weights according to their relative relevance. This policy is synchronic, periodic, and on-line and makes narrow changes.

Zhang and Yang (1999) propose a method for continually updating a feature-weighting scheme based on interactive user responses to the system's behavior. This is synchronic, conditional (on receiving user feedback), on-line, and broad.

Adaptation Maintenance. Leake, Kinley, and Wilson (1996) present an internal CBR approach to domain-level case adaptation in the DIAL system for disaster response planning. If adaptation is required to apply a response plan, DIAL first checks for applicable adaptation cases. When no adaptation cases apply, new adaptation cases can be learned by recording traces of rule-based or interactive manual adaptation. This type of adaptation learning is synchronic, based on the system's current adaptation knowledge (i.e., if there are problems, the rule-based or manual adaptation mechanism is invoked). The timing is conditional (if the adaptation is unable to be made automatically). The activation is on-line, during case-based planning for disaster response. The scope is narrow (it applies to one adaptation case).

Hanney and Keane (1996) describe a mechanism for learning adaptation rules by induction from differences in case knowledge. If two cases differ in only a small number of attributes, then the differences in those attributes can form the basis of adaptation rules from one context to another. If there are consistent adaptation types found among potential subsets of cases, then the type of difference can be learned as adaptation rule. This policy is synchronic, ad hoc, off-line, and broad.

Vocabulary Maintenance. Leake and Wilson (1999) describe DRAMA, an interactive CBR system for aerospace design that uses a proactive policy to help maintain the system vocabulary. The cases in the system are conceptual aircraft designs, for which the designers have a great deal of freedom in externalizing their design conceptualizations, freely defining new features to describe design cases. The proactive vocabulary policy examines the current design context and offers suggestions on appropriate concepts and relations that have been used previously. In this way the vocabulary is built in parallel with the case library. This policy is based on the current state of vocabulary knowledge and is synchronic. The timing is ad hoc, since the interactive nature of the system is under user control. The integration is on-line, and the scope is narrow.

Minor and Hanft (2000) describe an interactive framework that supports the maintenance of term dictionaries in parallel with the revision of case content. Terms are linked in with the current vocabulary (synchronic), at the user's request during processing (ad hoc and on-line), with narrow changes being made to the vocabulary.

6. COORDINATING MAINTENANCE ACROSS KNOWLEDGE CONTAINERS

The multiple knowledge containers of CBR overlap; knowledge available in one can replace missing knowledge in another (Richter 1998). As a result, just as builders of CBR systems can select the most convenient form in which to provide knowledge to an initial CBR system, maintainers of CBR systems can choose where to focus their maintenance efforts, applying effort where it is most convenient or effective. For example, the same overall effects on system accuracy might be achieved by case-base reorganization—which we consider part of CBM—or by adjustment of the similarity measure—which affects the similarity knowledge container. This raises two new issues for CBRM that do not arise at the individual knowledge container level: selection of the knowledge container to maintain and managing interactions between maintenance operations in different knowledge containers.

6.1. Selecting the Container to Maintain

When performance goals dictate the need for maintenance, a CBRM system must determine which knowledge container(s) to revise. In some situations, only one knowledge container will be an appropriate target, whereas in others multiple candidate revisions could be made. For example, failure to solve a problem could be addressed by adding a new case, adjusting similarity criteria (if the problem could have been solved starting from an existing case but that case was not retrieved), or adding adaptation knowledge. How to perform the credit assignment to identify which knowledge container to adjust has received some initial attention [e.g., Leake (1996) for identifying indexing problems versus missing cases] but is largely an open issue. When changes to multiple containers could be effective, utility-based choices may be needed to decide which container(s) to revise.

6.2. Managing Interactions between Knowledge Containers

Just as maintenance operations in one knowledge container may reduce the need for maintenance in others, maintenance in one container may necessitate maintenance in others as well. This may arise in either of two ways. First, some maintenance operations, such as revisions to the case representation vocabulary, intrinsically affect multiple knowledge containers. In order for the system to function, knowledge containers such as similarity and adaptation knowledge must be revised to handle the new representations. Thus vocabulary revisions to any container must be coupled with associated operations to adjust the other containers. Second, maintenance operations in one knowledge container may require adjustments to other containers in order to maintain performance or exploit revisions. For example, Leake et al. (1997) show that realizing the benefits of augmented adaptation knowledge may depend on associated revisions of similarity criteria in order to focus retrieval on appropriate cases for the revised adaptation knowledge. Heister and Wilke (1998) also provide a listing of the knowledge containers affected by the operations defined in their architecture.

Thus a general view of CBRM depends on three things: characterizing individual maintenance policies for specific knowledge containers, making strategic decisions about which strategies to apply, and characterizing how those policies are coordinated and integrated across knowledge containers to produce effective systemwide maintenance procedures.

7. METAMAINTENANCE

The analysis of maintenance policies makes clear that maintenance knowledge itself should be considered a knowledge container for CBR. Thus, like the other knowledge containers, it may require maintenance. Consequently, CBRM must include the capability for *metamaintenance*: maintenance of the maintenance strategies themselves. We can view the process of metamaintenance as a way to maintain the CBR system's knowledge containers indirectly. Instead of modifying the knowledge containers, this process puts in place or modifies the policies that will be used to modify them.

We illustrate metamaintenance with an example from our work on using metamaintenance to perform "lazy" updating of the case base in CBM (Leake and Wilson 1998). In standard CBR, when the system adapts a case to a new problem, the resulting case is stored, and the old case is left unchanged—it is assumed that the old solution still applies. However, sometimes part of the adaptation is performed because the old case no longer correctly solves the problem it solved previously (e.g., in a real estate price estimating system, because construction of a nearby shopping center affected real estate prices). In this case, an additional CBM policy can be put into place to revise each old case as it is being used. The result is to update old cases in a "lazy" manner as they are applied to new situations. This method is useful when environmental changes require general changes in the case base but global updating is not cost-effective, e.g., because updates are expensive and only a small portion of the case base is actually used.

Thus broad-scoped case-base updates can be done either directly by CBM making the change to all cases simultaneously, when it next performs maintenance, or by a narrow-scoped change to maintenance knowledge, adding a new maintenance rule to update each case that is retrieved, before it is applied to the new situation. This lazy approach may be preferable in resource-constrained circumstances.

8. DIRECTIONS FOR CBRM

Many CBRM issues remain to be investigated. Because, to our knowledge, the role of usage trends (diachronic type analysis) in guiding maintenance has not yet been explored in other research, we consider it an especially promising area. The very simple trend-based maintenance described in Leake and Wilson (1998) has application to a particularly well-behaved type of change in the case base that appears in other contexts as well (e.g., updating old prices based on inflation for real-estate appraisal) but would fail to apply to more subtle trends that would require more sophisticated methods.

Another form of trend information that might be exploited, for example, is patterns in the types of problems that are being solved. Examination of these patterns may identify "hot spots" in the problem space and determine subsets of the case base to be consulted first, whereas (if storage were limited) less useful cases could be archived (Leake and Wilson 1999b). Racine and Yang (1997) observe that recent cases may be likely to be useful; trend analysis could provide other types of suggestions for which cases should be most accessible.

Coordinating maintenance across knowledge containers is also an important area for future research. Despite considerable research on maintenance policies for individual knowledge containers, there has been comparatively little investigation of how to select containers to maintain and how policies that affect more than one knowledge container interact with one another. Deciding which knowledge container to maintain and how to do so in order to address performance goals will be important in managing more complex maintenance agendas. One interesting area is knowledge container transfer, moving knowledge from one container to another to locate knowledge where it can be used most easily and effectively. Shiu et al. (2000) describe a method for transfer from the case base to adaptation knowledge.

Reactive maintenance has received a great deal of attention, but the reactive approach presumes that there has been some type of system failure or critical condition to which the policies are reacting. This implies that reasoning in reactive situations will not meet performance goals or will not meet them as well. Proactive maintenance, in which policies anticipate maintenance needs before failures occur and proactively make changes, can help provide a more uniform way to continue meeting performance goals.

To exploit advances in CBRM, the advances must be accompanied by increased understanding of how to apply them. A long-term goal of our work on characterizing maintenance policies is to combine the characterizations with descriptions of the tasks, domains, and performance objectives for which particular policies are likely to be appropriate to help guide policy selection decisions when developing CBR systems.

9. CONCLUSION

This article presents a general framework for characterizing knowledge container maintenance policies in case-based reasoners. It presents basic dimensions for CBRM policies in terms of three subprocesses—data collection, triggering, and execution—and characterizes key design choices in terms of those dimensions. Factors considered include the type of information collected, timing, and integration of data collection; the timing and integration of maintenance triggering; whether the approach is reactive or proactive; the types of maintenance operations used; and the timing, integration, and scope of maintenance execution. The article demonstrates the use of this framework to describe multiple maintenance approaches. Further examination of the general maintenance task is clearly needed both to refine our understanding and to guide the development of CBRM practice. We hope that the framework presented in this article will spark further investigation both of maintenance practice and of issues and opportunities for new CBRM approaches.

ACKNOWLEDGMENTS

This research was supported in part by NASA under Award No. NCC 2-1035.

REFERENCES

AHA, D. W., and L. BRESLOW. 1997. Refining conversational case libraries. *In* Proceedings of the Second International Conference on Case-Based Reasoning. Springer-Verlag, Berlin, pp. 267–278.

AHA, D., D. KIBLER, and M. ALBERT. 1991. Instance-based learning algorithms. Machine Learning, **6**:37–66.

BHATTA, S., and ASHOK GOEL. 1995. Model-based indexing and index learning in analogical design. *In* Proceedings of the Seventeenth Annual Conference of the Cognitive Science Society. Lawrence Erlbaum, Mahwah, NJ, pp. 527–532.

BORRON, J., D. MORALES, and P. KLAHR. 1996. Developing and deploying knowledge on a global scale. *In* Proceedings of the Thirteenth National Conference on Artificial Intelligence, Vol. 2. AAAI Press, Menlo Park, CA, pp. 1443–1454.

CHEETHAM, W., and J. GRAF. 1997. Case-based reasoning in color matching. *In* Proceedings of the Second International Conference on Case-Based Reasoning. Springer-Verlag, Berlin, pp. 1–12.

DEANGDEJ, J., D. LUKOSE, E. TSUI, P. BEINAT, and L. PROPHET. 1996. Dynamically creating indices for two million cases: A real world problem. *In* Advances in Case-Based Reasoning. *Edited by* I. Smith and B. Faltings. Springer-Verlag, Berlin, pp. 105–119.

DIETTERICH, T. 1986. Learning at the knowledge level. Machine Learning, **1**:287–316.

DOMINGOS, P. 1995. Rule induction and instance-based learning. *In* Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence. Morgan Kaufmann, San Francisco, CA, pp. 1226–1232.

DOYLE, MICHELLE, and PÁDRAIG CUNNINGHAM. 1999. On balancing client-server load in intelligent web-based applications involving dialog. Technical Report TCD-CS-1999-25, Trinity College, Dublin.

FOX, S., and D. LEAKE. 1995. Modeling case-based planning for repairing reasoning failures. *In* Proceedings of the 1995 AAAI Spring Symposium on Representing Mental States and Mechanisms. AAAI Press, Menlo Park, CA, pp. 31–38.

FRANCIS, A., and A. RAM. 1993. Computational models of the utility problem and their application to a utility analysis of case-based reasoning. *In* Proceedings of the Workshop on Knowledge Compilation and Speed-Up Learning.

GÖKER, MEHMET, and THOMAS ROTH-BERGHOFER. 1999. Development and utilization of a case-based help-desk support system in a corporate environment. *In* Proceedings of the Third International Conference on Case-Based Reasoning. *Edited by* K. D. Althoff, R. Bergmann, and L. K. Branting. Springer-Verlag, Berlin, pp. 132–146.

HAMMOND, K. 1989. Case-Based Planning: Viewing Planning as a Memory Task. Academic Press, San Diego.

HANNEY, K., and M. KEANE. 1996. Learning adaptation rules from a case-base. *In* Proceedings of the Third European Workshop on Case-Based Reasoning. Springer-Verlag, Berlin, pp. 179–192.

HANNEY, K., M. KEANE, B. SMYTH, and P. CUNNINGHAM. 1995. What kind of adaptation do CBR systems need? a review of current practice. *In* Proceedings of the Fall Symposium on Adaptation of Knowledge for Reuse. AAAI Press, Menlo Park, CA.

HEISTER, F., and W. WILKE. 1998. An architecture for maintaining case-based reasoning systems. *In* Proceedings of the Fourth European Workshop on Case-Based Reasoning. *Edited by* P. Cunningham, B. Smyth, and M. Keane. Springer-Verlag, Berlin.

IHRIG, L., and S. KAMBHAMPATI. 1997. Storing and indexing plan derivations through explanation-based analysis of retrieval failures. Journal of Artificial Intelligence Research, **7**:161–198.

KITANO, H., and H. SHIMAZU. 1996. The experience sharing architecture: A case study in corporate-wide case-based software quality control. *In* Case-Based Reasoning: Experiences, Lessons, and Future Directions. *Edited by* D. Leake. AAAI Press, Menlo Park, CA, pp. 235–268.

LEAKE, D. 1996. Experience, introspection, and expertise: Learning to refine the case-based reasoning process. Journal of Experimental and Theoretical Artificial Intelligence.

LEAKE, D., and D. WILSON. 1998. Case-base maintenance: Dimensions and directions. *In* Proceedings of the Fourth European Workshop on Case-Based Reasoning. *Edited by* P. Cunningham, B. Smyth, and M. Keane. Springer-Verlag, Berlin, pp. 196–207.

LEAKE, D., and D. WILSON. 1999a. Combining CBR with interactive knowledge acquisition, manipulation and reuse. *In* Proceedings of the Third International Conference on Case-Based Reasoning. Springer-Verlag, Berlin, pp. 203–217.

LEAKE, D., and D. WILSON. 1999b. When experience is wrong: Examining CBR for changing tasks and environments. *In* Proceedings of the Third International Conference on Case-Based Reasoning. Springer-Verlag, Berlin, pp. 218–232.

LEAKE, D., and D. WILSON. 2000. Remembering why to remember: Performance-guided case-base maintenance. *In* Proceedings of the Fifth European Workshop on Case-Based Reasoning. *Edited by* E. Blanzieri and L. Portinale. Springer-Verlag, Berlin,

LEAKE, D., A. KINLEY, and D. WILSON. 1996. Acquiring case adaptation knowledge: A hybrid approach. *In* Proceedings of the Thirteenth National Conference on Artificial Intelligence. AAAI Press, Menlo Park, CA, pp. 684–689.

LEAKE, D., A. KINLEY, and D. WILSON. 1997. Learning to integrate multiple knowledge sources for case-based reasoning. *In* Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence. Morgan Kaufmann, San Mateo, CA, pp. 246–251.

MINOR, MIRJAM, and ALEXANDRE HANFT. 2000. Corporate knowledge editing with a life cycle model. *In* Proceedings of the Eighth German Workshop on Case-Based Reasoning.

MUÑOZ-AVILA, H. 1999. A case retention policy based on detrimental retrieval. *In* Proceedings of ICCBR-99.

MUÑOZ-AVILA, H., and J. HUELLEN. 1996. Feature weighting by explaining case-based planning episodes. *In* Proceedings of the Third European Workshop on Case-Based Reasoning. Springer-Verlag, Berlin, pp. 280–294.

NETTEN, B. D. 1999. Verification of case-base integrity in BRIDGE. *In* Proceedings of the Seventh German Workshop on Case-Based Reasoning. pp. 120–130.

PORTINALE, L., P. TORASSO, and P. TAVANO. 1999. Speed-up, quality, and competence in multi-modal reasoning. *In* Proceedings of the Third International Conference on Case-Based Reasoning. Springer-Verlag, Berlin, pp. 303–317.

RACINE, K., and Q. YANG. 1997. Maintaining unstructured case bases. *In* Proceedings of the Second International Conference on Case-Based Reasoning. Springer-Verlag, Berlin, pp. 553–564.

REINARTZ, T., I. IGLEZAKIS, and T. ROTH-BERGHOFER. 2000. On quality measures for case base maintenance. *In* Proceedings of the Fifth European Workshop on Case-Based Reasoning. *Edited by* E. Blanzieri and L. Portinale. Springer-Verlag, Berlin,

RICHTER, M. 1998. Introduction. *In* CBR Technology: From Foundations to Applications. *Edited by* M. Lenz, B. Bartsch-Spörl, H.-D. Burkhard, and S. Wess. Springer-Verlag, Berlin, Chap. 1, pp. 1–15.

SHIMAZU, H., and Y. TAKASHIMA. 1996. Detecting discontinuities in case-bases. *In* Proceedings of the Thirteenth National Conference on Artifical Intelligence, Vol. 1. AAAI Press, Menlo Park, CA, pp. 690–695.

SHIU, S., C. SUN, X. WANG, and D. YEUNG. 2000. Maintaining case-based reasoning systems using fuzzy decision trees. *In* Proceedings of the Fifth European Workshop on Case-Based Reasoning. *Edited by* E. Blanzieri and L. Portinale. Springer-Verlag, Berlin,

SMYTH, B. 1998. Case-base maintenance. *In* Proceedings of the Eleventh International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, Castellon, Spain.

SMYTH, B., and P. CUNNINGHAM. 1996. The utility problem analysed: A case-based reasoning perspective. *In* Proceedings of the Third European Workshop on Case-Based Reasoning. Springer-Verlag, Berlin, pp. 392–399.

SMYTH, B., and M. KEANE. 1995. Remembering to forget: A competence-preserving case deletion policy for case-based reasoning systems. *In* Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence. Morgan Kaufmann, San Francisco, pp. 377–382.

SMYTH, B., and E. MCKENNA. 1999a. Building compact competent case-bases. *In* Proceedings of the Third International Conference on Case-Based Reasoning. Springer-Verlag, Berlin.

SMYTH, B., and E. MCKENNA. 1999b. Footprint-based retrieval. *In* Proceedings of the Third International Conference on Case-Based Reasoning. Springer-Verlag, Berlin.

SMYTH, B., and E. MCKENNA. 2000. An efficient and effective procedure for updating a competence model for case-based reasoners. *In* Proceedings of the Eleventh European Conference on Machine Learning. Springer-Verlag, Berlin.

SURMA, JERZY, and JANUSZ TYBURCY. 1998. A study on competence-preserving case replacing strategies in case-based reasoning. *In* Proceedings of the Fourth European Workshop on Case-Based Reasoning. *Edited by* P. Cunningham, B. Smyth, and M. Keane. Springer-Verlag, Berlin, pp. 233–238.

VOß, ANGI 1996. Principles of case reusing systems. *In* Advances in Case-Based Reasoning. *Edited by* I. Smith and B. Faltings. Springer-Verlag, Berlin, pp. 428–444.

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

WATSON, IAN, and DAN GARDINGEN. 1999. A distributed case-based reasoning application for engineering sales support. *In* Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, Vol. 1. Morgan Kaufmann, San Mateo, CA, pp. 600–605.

WETTSCHERECK, D., D. AHA, and T. MOHRI. 1997. A review and empirical evaluation of feature-weighting methods for a class of lazy learning algorithms. Artificial Intelligence Review, **11**(1–5):273–314.

YANG, QIANG, and JING WU. 2000. Keep it simple: A case base maintenance policy based on clustering and information theory. *In* Proceedings of the Canadian AI Conference.

ZHANG, ZHONG, and QIANG YANG. 1999. Dynamic refinement of feature weights using quantitative introspective learning. *In* Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence. Morgan Kaufmann, San Mateo, CA.

ZHU, JUN, and QIANG YANG. 1999. Remembering to add: Competence-preserving case-addition policies for case base maintenance. *In* Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence. Morgan Kaufmann, San Mateo, CA.