# Case-Based Reasoning for Gas Turbine Diagnostics

**Mark Devaney[1], Bill Cheetham[2]**

[1]Enkia Corporation, 10437 Innovation Drive, Wauwatosa, WI, 53226, USA, *markd@enkia.com*
[2]General Electric Global Research, 1 Research Circle, Niskayuna, NY, 12309, USA, *cheetham@crd.ge.com*

**Abstract**

General Electric used case-based reasoning for gas turbine diagnostics at their monitoring and diagnostics center in Atlanta, GA. This application had requirements that included accuracy, maintainability, modularity, parameterization, robustness, and integration of the system into an existing infrastructure. The CBR system has a modular "plug and play" architecture to facilitate experimentation and optimization. It was integrated into the production environment in 2004. The CBR system is currently in a trial deployment where diagnoses made by the system are created along with the previous process of using human-generated diagnosis.

## Introduction

This paper describes a Case-Based Reasoning (CBR) system (Kolodner 1993, Aamodt and Plaza 1994) to help General Electric (GE) Energy service the heavy-duty gas turbines it sells. Figure 1 shows a typical turbine, which can be used to power a small city and costs $30 million. As of 2004 there are over 6,000 of these turbines in use by GE customers worldwide. GE has contracts to service 1,200 turbines and that number has been growing by hundreds every year. The goals of the service are to improve turbine and system reliability, reduced turbine operating/maintenance costs, and produce the greatest possible sustained availability from the power generation equipment. All turbines have control logic, which includes the ability to automatically shut down the turbine to protect it from potentially dangerous situations. An unplanned automatic shutdown is called a *trip*. When a turbine that is serviced by GE trips, people at GE's monitoring and diagnostics (M&D) center remotely diagnose the problem and assist the customer in quickly restarting the turbine. This paper describes the CBR system that was created for GE's M&D center.

### Related Work

GE has a long history of using CBR for monitoring and diagnostics. Two of these efforts are for medical equipment and locomotives. An Error Log Similarity Index (ELSI) (Cuddihy and Cheetham 1999) was created in 1992 to solve a diagnostic challenge with computer tomography scanners. GE had built an infrastructure to allow service to dial in to imaging equipment and diagnose failures while still on the phone with the customer. Once the service engineer had taken a look at the equipment remotely, the customers' issues could often be resolved over the phone. The cases consisted of error logs generated by the computer processes controlling the machines. ELSI has been in continuous use since 1992.

CBR was also used for remote diagnostics on locomotives to be able to quickly identify if an equipment failure has already occurred, or is going to occur on-board the locomotive that may result in a locomotive stranded on the tracks. The cases consisted of historical fault logs generated by the locomotive and the repair history of the locomotive. The system (Varma and Roddy, 1999) has been in continuous use since 1995.

Other companies have used artificial intelligence techniques for diagnosing gas turbines. One significant system is TIGER (Milne and Nicol, 2000). The purpose of TIGER is to identify future trips as early as possible or diagnose the cause of unexpected events. It uses rule-based and model-based reasoning in its monitoring and diagnosis.

TIGER was employed by groups at GE Energy for diagnosis, but the thousands of rules used by the system were difficult to understand and modify. These concerns over maintainability were a large motivation for the use of case-based reasoning which has the potential to greatly reduce the amount of effort required to maintain an active production system in a large-scale deployment environment.

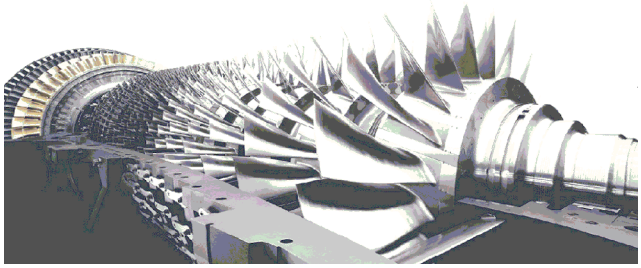

**Figure 1: The Inside of a Gas Turbine**

### Turbine Diagnosis Process

Gas turbines have controllers at the site where they are installed that perform a variety of functions including protecting the turbine from damage by performing an

automated shutdown (trip) when dangerous conditions are detected and archiving sensor data. Typical gas turbines will have about 2000 sensors providing information such as temperatures, pressures, and vibration levels every second. If the turbine is serviced by GE, there will also be a remote connection to send the data to GE's M&D center in Atlanta.

Prior to the development of the CBR system, the turbine diagnosis process was as follows. When a turbine trips, the on-site controller will send a message to Atlanta. The trip will be assigned to a person in the M&D Center for analysis. That person will access the data from the turbine, review key values, create a hypothesis about the trip cause, create plots specific to the trip type hypothesized, confirm the cause of the trip as best as can be done using the available data, then call the site that tripped to provide assistance and confirm the trip cause. The advice is most valuable soon after the trip, so the M&D Center tries to analyze each trip in under ten minutes.

The CBR system is used to automate the data review, hypothesis generation, and hypothesis confirmation portions of this process whenever possible and assist the user when it does not have confidence in a single cause. This paper will describe how the CBR system was created and how it achieves this goal.

## System Requirements

As with any deployed software application, the first steps in creating the system involve gathering requirements from the customer and assessing the conditions and environment under which the system will be used. Given our team's experience with Case-Based Reasoning and other AI techniques and the maturity of the processes at GE's M&D center, the project began with a relatively well-defined set of requirements on the accuracy, maintainability, modularity, parameterization, robustness, and integration of the system.

### Accuracy

The desire for diagnostic accuracy is no surprise, but there was also a need for the CBR system to not make diagnoses when it did not have sufficient confidence in its hypothesis. This is particularly important in "human in the loop" applications such as the one described here, where operators are being guided by the output of the intelligent system. Due to the time spent trying to validate inaccurate predictions, it is preferable that the system take a conservative approach to prediction and make no recommendations when it is not sufficiently confident of its answer (Cheetham and Price, 2004). This goal required techniques to represent and quantify the tradeoffs of various predictive outcomes.

### Maintainability

One of the strengths of case-based reasoning as a solution to the diagnosis problem is its potential for reducing the amount of human effort required for creating and maintaining the application, primarily in the knowledge engineering/acquisition stages (Pal & Shiu, 2004). Rule or model-based approaches often require very extensive knowledge engineering efforts by experts versed in the appropriate techniques in conjunction with subject matter experts in order to construct an application. Furthermore, once that application is fielded in a dynamic changing real-world environment these techniques require the same personnel to monitor and update the system in order to preserve an acceptable level of performance. Case-based reasoning, in contrast, is amenable to automated maintenance, in which a "meta-reasoning" component of the architecture continually monitors the inputs and outputs of the system and makes modifications to the casebase autonomously in order to preserve currency with a changing application domain.

### Modularity

We are in agreement with the view that case-based reasoning is a methodology as opposed to a technology (Watson, 1998). In this view, the "what" of the CBR process of "retrieve-reuse-revise-retain" is implemented by various algorithms – the "how". By employing a modular architecture with well-defined component interfaces, experimentation with different implementations and/or different algorithms for the various stages of CBR is greatly facilitated. This was particularly important for this project, which represents a collaboration among several different groups and organizations, each taking responsibility for different stages of the process at different times. The modular architecture allowed these different groups to include their particular "how" into the overall system and evaluate the resulting performance without having to deeply understand the underlying workings of the system or the particulars of the other components in the overall system.

### Parameterization

Many of the techniques typically employed in case-based reasoning systems are controlled by various parameters. Typically, as part of a system deployment, knowledge engineers perform experimentation and determine an appropriate set of values for these parameters which are then fixed. The danger in this approach is that while the initial set of parameter values may be optimal given the data available at system design and implementation time, a system that is to be deployed in a dynamic and changing environment must be able to respond to these changes via alteration of internal parameterization. In other production deployment scenarios, while the underlying domain may not change significantly over time, only a limited sample of data may be available at the time the system is constructed and deployed. As this data becomes available over time it is desirable to easily modify internal parameters in response to a larger sample of the population.

Our CBR application was designed in such a way that operational parameters of the system could be changed without requiring modification of the code. This allowed us to use tools and methodologies for automated optimization such as GE's design of experiments (DOE) software and genetic algorithm decision engine for searches of large configuration spaces (Aggour, et. al., 2003). We were able to use these tools to not only automatically determine an optimal configuration before deploying the application, but plan to incorporate these methods into an ongoing automatic maintenance and optimization strategy using the cross-validation capabilities of the CBR system.

### Robustness

One of the greatest challenges in applying Artificial Intelligence techniques to real-world problems in real-world environments is the presence of noisy and/or incomplete data. In this application of turbine failure diagnosis, the data values used to create cases and describe new incidents come from physical hardware sensors onboard the turbines. This data must be collected onsite and transmitted to the M&D facility in Atlanta. It is quite common for particular sensors to have missing data due to hardware failures, data collection issues, or transmission errors. Furthermore, because a common representation is used to cover a variety of different turbines, there are attributes that are only present in certain turbine models, which introduces another source of missing data. Noise in the sensor values can occur for similar reasons – sensor malfunction, data transmission or collection issues. Different sites may also run their turbines with different operating parameters, causing some sensor readings to be outside of normal expected ranges.

In response to these issues, the algorithms employed by the CBR system must be robust in the face of missing and/or erroneous data – it is unrealistic to assume that "prototypical" cases or exemplars will exist or that newly-arriving failure incidents to be diagnosed will be described with the full set of available attributes. Finally, due to the degree of subjectivity involved in assigning case labels to incidents of turbine failures, there is a lack of uniformity in the case labels (i.e., the reason for the trip). These labels are assigned by human troubleshooting operators who often have conflicting hypotheses as to the true root cause of some failures, leading to issues with case and incident labels not corresponding to the attributes that describe them. This leads to interesting issues that must be confronted in the hypothesis generation phase of the CBR process as well as the types of self-learning and evaluation strategies that can be employed during the maintenance process.

### Integration

The final major requirement of the application was the ability to be easily integrated into GE's central monitoring platform in a production environment. While this requirement typically impacts the software engineering aspect of the application as opposed to the knowledge engineering aspect, it is equally critical in contributing to a successful deployment. Our use of industry standard technologies for integration into a production environment such as XML and JDBC as well as our techniques for testing and evaluation are discussed later.

## Knowledge Engineering

Knowledge engineering is the process by which the system designers determine what information is necessary to produce the desired outputs of the intelligent system and determine the algorithms or techniques to make use of that data. Adequate representations are a cornerstone of a successful application of intelligent systems techniques. In our domain of turbine diagnostics, the universe of available representations was effectively limited to the outputs of the hardware sensors monitoring the turbines. This is the same data that the human M&D engineers use to make diagnoses and thus, presumably, carries information sufficient for the CBR system to perform the same task. Consultation with M&D engineers resulted in lists of commonly-examined sensors for each type of trip, which were validated and standardized to account for differences of opinion among different engineers.

Use of this "raw" low-level data may be a viable strategy and has the benefit of greatly reducing the amount of effort required in the knowledge engineering task – it is often simply a matter of collecting and organizing the raw data. The drawback to using this type of data, however, is that due to its representational "sparseness", it is generally necessary to collect a greater population sample in order to construct a casebase with adequate coverage. In other words, the use of raw low-level attributes typically involves a much higher-dimensional representation space and the values of those attributes typically have much larger ranges, further increasing the size of the space that the casebase must cover.

At the other representational extreme, human experts or other external knowledge sources, or even automated induction, clustering, or other techniques are used to construct a more restricted description space that carries the same informational content. The tradeoff in this case is the extra effort required to construct the representations. Another tradeoff is that this effort must be repeated whenever it is desired to expand the role of the system to additional diagnosis types, application conditions, etc.

In this application, however, it was decided that due to the large number of low-level features present, particularly in respect to the number of historical cases available, it was preferable to employ knowledge-based representations by extracting higher-level features from the low-level data. This process was carried out by subject matter experts in consultation with AI knowledge engineers. The large number of raw sensor tags were distilled into a more parsimonious set of attributes while still capturing the predictive information. This set of attributes was used as

the format for a case. For example, thirty minutes of control data can be summarized by what the last control change was and when it happened. This knowledge engineering process was somewhat of a middle ground, however, as a fairly "coarse" set of higher-level attributes was defined rather than a perfectly fitting set of finer-grained attributes. We then employed optimization and analysis tools, such as Weka (Witten and Frank, 2000), for automated feature selection to arrive at the optimal set of descriptors given our training data.

After defining a coarse set of derived features and determining the optimal subset of these features via automated analysis, the final stage of the initial knowledge engineering phase of the project was to systematically evaluate various parameterizations, such as feature weights and configurations of the complete system using cross-validation studies and a genetic algorithm on our training casebase. The initial training data consisted of historical trip records that were validated by human experts. We later validated the parameters on a new set of trip instances obtained after the system had been built. The optimization and evaluation phase continued through the full development cycle. The results of the ongoing evaluations were used during system design and implementation in order to identify techniques and algorithms that prove most suitable for the problem as well as highlight aspects of the system where parameterization was desirable.

Thus, knowledge engineering is not necessarily a discrete first step as is requirements gathering and analysis (at least to a degree), but a continuous ongoing process through the life of the project. In fact, many of the techniques and algorithms developed and employed during the knowledge engineering phase of the project not only contributed to the initial system design and implementation, but have been identified as candidates for inclusion in the ongoing automated maintenance of the system.

## Case-Based Reasoning Architecture

Our CBR architecture for turbine diagnostics follows the traditional case-based reasoning paradigm of "retrieve-reuse-revise-retain" using numerical reasoning methods to accommodate the data available in the application domain.

As mentioned above, the CBR system reflects a modular "plug and play" architecture to facilitate experimentation and optimization. Figure 2 depicts the high-level stages of reasoning in our implementation but does not reflect an actual software architecture – i.e., it is a depiction of functional as opposed to physical configuration. The following sections describe these stages in more detail.

### Input

When a turbine trip report is received for diagnosis, its low-level description is a time-series of sensor values read from some interval prior to failure. These sensor values are converted to a higher-level representation via the knowledge-based feature extraction algorithms described above. The output of this process is the input to the CBR system, and all of this activity occurs automatically without human intervention when a trip is activated.

### Configuration

The case-based reasoning system is configured via a set of XML files that control the composition and behavior of the entire system. This configuration specifies which "plug and play" modules should be used for retrieve, recommend, and confidence. It also includes all parameters needed for each module, such as the attribute weights that will be used by the retrieve module to calculate the similarity of a case. Retuning the parameters in the configuration files is one aspect of periodic maintenance. Another configuration file specifies the location and structure of the case library as well as the representations used in describing cases.

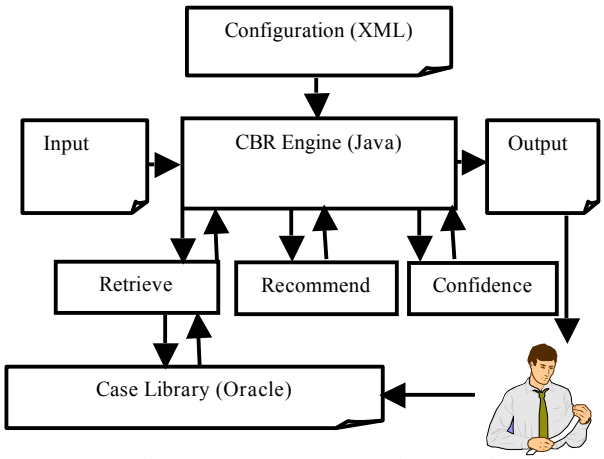


**Figure 2: Case-based Reasoning System Architecture**

### CBR Engine

The top-level CBR Engine reads the configuration files, creates and configures all of its components and loads the casebase into memory. The CBR Engine provides the top-level interface of the overall system for diagnosing new incidents and monitoring the behavior of the system and is the component responsible for executing each of the modules that compose the overall system.

### Retrieve

The first step in the CBR process is the retrieval of similar prior trip events. In this step the case library is queried for prior events that are most similar to the current incident to be diagnosed. A custom nearest-neighbor technique is used to retrieve some number of cases that form the input of the next stage in reasoning.

### Recommend

In traditional descriptions of case-based reasoning, prior cases are "adapted" to form solutions to new problems. In

situations where CBR is used in a multi-step problem solving or planning task, this adaptation may involve combining the steps or other aspects of one or more cases to produce a novel solution (e.g., as in Hammond, 1996). In this application domain, however, the output of the CBR system must be a single diagnosis, which comes from a discrete set of potential candidates. A weighted voting algorithm is used to combine the solutions suggested by the recalled cases into a single answer. The requirement to produce only a single solution or small set of potential solutions is not unusual in recommendation systems. Concern for usability often dictates that the system not force the user to manually investigate multiple potential answers. Instead, a single answer along with the system's confidence in that answer should be presented and when the system does not have adequate confidence in any solution it should present that information instead. I.e., no answer is often preferable to an incorrect answer.

### Confidence

As described above, one of the critical factors of the CBR system is the requirement that it provide a confidence level along with its recommendation to the M&D support personnel. After the recommendation is determined, the next stage of the CBR process assigns a confidence level to that recommendation. In our application, a histogram is computed based on the weighted recommendations of the supporting cases. The histogram is normalized and each recommendation is assigned a confidence based on its corresponding histogram value. The output of the confidence step is the diagnosed root cause of the new turbine failure along with a confidence assessment of that diagnosis.

### Output

One of the most critical steps in case-based reasoning is the fourth "R" – retain. By incorporating the problems solved into the casebase, a CBR system is able to learn from its experiences and improve its performance over time as it gains additional experience. This ability is particularly important in a fielded diagnostics application such as this one, where the reasoning system must maintain currency with changing conditions in the domain in which it is operating. GE is continually introducing modifications, new models, new parts, and other changes into its deployed fleet and the automated diagnostic system must be able to constantly update its knowledge base in order to reflect the effects of these changes. The relative ease with which case-based reasoning is able to function in this type of dynamic changing environments is one of its principal strengths with respect to model-based or rule-based approaches (e.g., "expert systems"). In these approaches, human experts must be called in to incorporate the new information, whereas in CBR, it is largely a matter of simply adding the new data points into the case library.

## Deployment and Evaluation

After finalizing the software architecture and implementation, the CBR system was parameterized and optimized via a combination of "hand tuning" and automated analysis via GE's genetic algorithm engine and DOE software as mentioned above. During these processes, a data set of approximately four hundred trips was used and repeated cross validation studies were performed. This data set would also be used as the casebase once the system was deployed into production at the M&D center.

Due to the criticality of the case labels, extensive human validation was performed by asking subject matter experts to examine all cases that were misclassified during the cross validation studies. Due to differences of opinion between different experts as well as the inability to review raw tag values after some time, there was still not complete confidence in the validity of all case labels. However, this situation is not unusual for real-world applications and our robust retrieval and recommendation algorithms were able to obtain performance in excess of the performance requirements.

In order to reduce the effort required to validate the trip incidents to be used in the casebase, only cases that were misdiagnosed during the cross-validation studies were examined. Some of these cases had their case labels corrected based on expert feedback and others were removed if the experts could not agree on the label. The remaining cases were retained as their misdiagnoses were attributed to lack of coverage in the case base, a condition that is expected to be remedied as the system is placed in production and additional trips are processed and incorporated into the casebase.

After the final evaluations and configuration were complete, the system was integrated into the production environment in 2004. This entailed incorporating the system within the M&D troubleshooting workflow to invoke CBR whenever a new trip was received and to display the system's diagnosis to the human operator. The existing workflow system was modified to trigger the CBR engine via a daemon process and the CBR system received the new trip incident data from the database system via JDBC. The recommended diagnosis from CBR was incorporated into the web-based user interface already in use by the M&D personnel. In order to support ongoing maintenance of the system as well as monitor its performance, all diagnoses made by the system are logged along with the "official" human-generated diagnosis.

The system was placed in limited production use for a short period in order to verify full functionality, perform usability studies, and collect an initial sample of performance data. This data is being used to identify potential areas of improvement in the user interface and to perform blame assignment on misdiagnoses (as identified by discrepancies between the system and human opinions). These initial results have been encouraging and have also

spurred the development of automated techniques for casebase maintenance and blame assignment.

## Conclusions and Future Directions

As mentioned previously, our initial performance results have been encouraging and the case base is currently being augmented with a larger number and range of cases. The system currently performs on par with other approaches such as rule-based classification but does not require the same degree of maintenance. We are particularly encouraged by the degree to which the ease of maintenance requirement has been met by the system. Non-expert GE personnel have successfully increased the coverage of the system simply by adding cases for new failure types into the case library. The CBR system is able to successfully diagnose those root causes with as good or better accuracy than the ones initially used to create the case library.

Our immediate next step is to deploy the system in production for a full-scale test now that its stability and performance have been demonstrated in the trial deployment. The system will be run in parallel with the current human troubleshooting operators in the M&D center who will not see the CBR diagnosis during this stage. Nor will the CBR system have access to the cases processed during this stage. We expect to collect at least three months of data and to then begin evaluating our techniques for automated blame assignment and casebase optimization. These studies will be made offline using the newly collected data and a new casebase put into place. Eventually the ongoing maintenance and casebase optimization will be gradual transitioned to primarily online automated mode. At that time we will have "closed the loop" and have deployed a self-learning diagnostic system and will continue to monitor and collect performance statistics.

In addition to expanding our work into techniques and tools for automated maintenance of the CBR system, we are also exploring the potential for use of CBR in other aspects of turbine maintenance such as the prediction of trips before they occur. We are very pleased with the performance of CBR as a decision aid for troubleshooting turbine failures and are excited about its potential in other areas as well.

## Acknowledgements

The authors would like to point out that the work described in this paper was a team effort that included many people. We would like to thank Slavek Zaremba, Betty Whitaker, Bob Simpson, Sairam Satya-Neelan, Ashwin Ram, Joe Price, Janet Kolodner, and Rahul Chadha for their many contributions to this project. The authors also thank the anonymous FLAIRS reviewers for their many helpful and insightful comments.

## References

Aamodt, A., Plaza, E. 1994. Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches, AICOM, Vol. 7, No. 1.

Aggour, K., Pavese, M., Bonissone, P., Cheetham, W., 2003. SOFT-CBR: A Self-Optimizing Fuzzy Tool for Case-Based Reasoning, The 5th International Conference on Case-Based Reasoning, Trondheim, Norway, June 23 - 26.

Cheetham, W., Price, J., 2004. Measures of Solution Accuracy in Case-Based Reasoning Systems, Seventh European Conference on Case-Based Reasoning, Madrid, August 30 - September 2.

Cuddihy, P., Cheetham, W. 1999. ELSI: A Medical Equipment Diagnostic System. *Third International Conference on Case-Based Reasoning* 415-425. Munich, Germany.

Hammond, K.J. 1996. Chef: A model of case-based planning. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI-96)* 261-271. Portland, Oregon.

Kolodner, J., 1993. *Case-Based Reasoning*. Morgan Kaufmann: San Mateo, CA.

Milne, R., Nicol, C. 2000. Tiger™: Continuous Diagnosis of Gas Turbines. In Proc. Of ECAI/PAIS'00, edited by Werner Horn, Berlin, 2000.

Pal, S., Shiu, S. 2004. *Foundations of Soft Case-Based Reasoning*. Wiley-Interscience: Hoboken, NJ.

Varma, A., Roddy, N. 1999. ICARUS: A Case-Based System for Locomotive Diagnostics. *Engineering Applications of Artificial Intelligence Journal*.

Watson, I., 1998. Is CBR a Technology or a Methodology?. In, Tasks & Methods in Applied Artificial Intelligence. DelPobil, A.P., Mira, K., & Ali, M. (Eds.), pp525-534. Springer-Verlag Lecture Notes in Artificial Intelligence 1416, Berlin.

Witten, I., Frank, E., 2000. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann: San Francisco, CA.