AI Magazine Volume 27 Number 3 (2006) (© AAAI)

Automating the Underwriting of Insurance Applications

Kareem S. Aggour, Piero P. Bonissone, William E. Cheetham, and Richard P. Messmer

■ An end-to-end system was created at Genworth Financial to automate the underwriting of long-term care (LTC) and life insurance applications. Relying heavily on artificial intelligence techniques, the system has been in production since December 2002 and in 2004 completely automates the underwriting of 19 percent of the LTC applications. A fuzzy logic rules engine encodes the underwriter guidelines and an evolutionary algorithm optimizes the engine's performance. Finally, a natural language parser is used to improve the coverage of the underwriting system.

With more than 130 years of history, 15 million customers, $98 billion in assets, and $11 billion in annual sales, Genworth Financial (GNW) is one of the world's oldest and largest insurance providers. GNW is committed to providing financial protection to its customers, their families, and their businesses. This is accomplished through a diverse set of products, including long-term care, term life, dental, disability, and mortgage insurance. Long-term care (LTC) insurance is used to cover significant medical costs, such as home nursing care, to protect the policyholder's assets through illness and old age. Term life insurance provides benefits to the living upon the death of the insured. This article focuses on the automation of the LTC underwriting process, but much of the material applies to term life underwriting as well.

As GNW receives LTC insurance applications, an individual referred to as an underwriter reviews each to determine whether the applicant should be approved for coverage. Based on the applicant's medical history, the underwriter assigns the applicant to a discrete risk category or declines the applicant altogether. The risk category dictates the premium to be paid for the insurance, making appropriate placement critical. Underestimating the risk would result in the applicant not paying enough to cover the financial risk incurred insuring that individual. Overestimating the risk would result in GNW not being price competitive and losing customers. Prior to this automation effort, this crucial underwriting process was entirely manual.

GNW chose to automate this process to improve consistency and reduce the number of defects. For legal reasons the decision-making process had to remain transparent, constraining the technologies that were used.

The next section describes the manual underwriting process. The new automated process is then discussed. Next, the use of artificial intelligence (AI) technology and the surrounding system are presented. Benefits of the new system are provided, followed by details on the system development, deployment, and maintenance. Finally, some conclusions and future work are presented.

Manual Underwriting Process

The LTC underwriting process begins when a paper application (APP) is completed by hand, faxed to GNW, and then scanned into an electronic data warehouse. Underwriters located

 ISSN 0738-4602

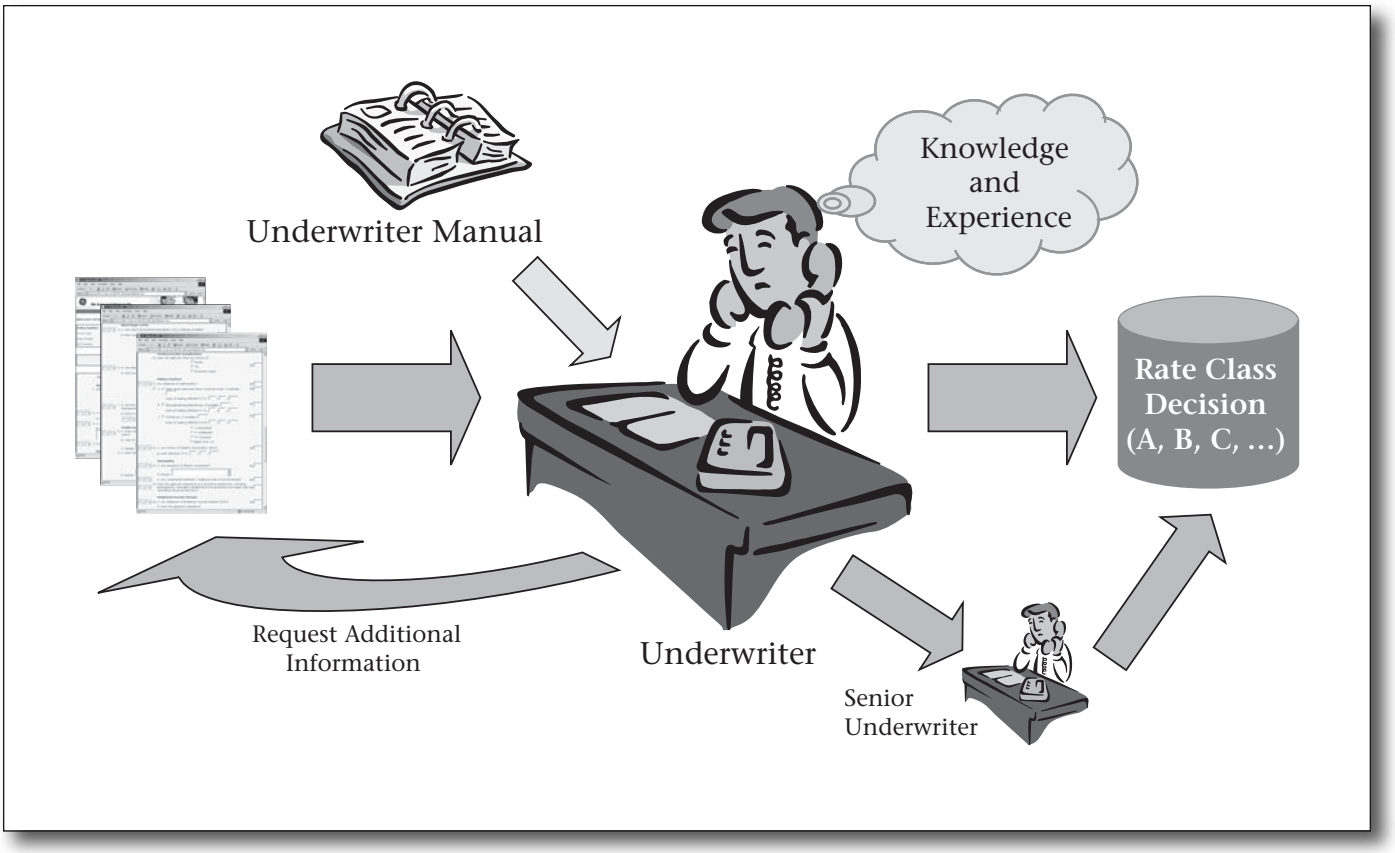


Figure 1. Manual Underwriting Process.

throughout the country view these scanned documents online, and then rate the risk of insuring each person. If the underwriter has any concerns, he can request additional information from the applicant through a phone health interview (PHI) or a face-to-face (F2F) interview, resulting in the submission of additional paper forms. At any time, an underwriter can also request an attending physician summary (APS)—a copy of the applicant's medical history from his or her primary physician. Before the automation of the underwriting process, volumes of these documents were ordered extraneously, providing no value at a great cost of time and money. One benefit of automation was reducing this waste.

Underwriters can make a decision at any point they feel they have sufficient information. If they have any questions or concerns, they can refer cases to a senior underwriter. Once a decision is made, the applicant is notified by mail. To evaluate the quality of the decisions produced, a percentage of the cases are randomly audited on a monthly basis. Figure 1 shows the manual process.

Underwriters make decisions following guidelines specified in an underwriter manual. They also rely upon extensive medical knowledge and personal experience when underwriting cases. The reliance upon their own experience and judgment causes inconsistency across the underwriters, resulting in inaccurate rate classifications. This use of personal knowledge and experience to make decisions also made this a difficult problem to automate.

Automated Underwriting Process

In automating the underwriting process, artificial intelligence techniques were used to codify the underwriter rules. These rules were then incorporated into a new, automated end-to-end rule-based system (Chisholm 2004). A fuzzy logic rules engine (FLRE) was designed and developed to codify the underwriter rules (Jang, Sun, and Mizutani 1997); this became the "digital underwriter" in the new process. This digital underwriter is able to determine whether an application should be sent to a human underwriter for review, allowing the automated process to be deployed without worrying about every possible case variation. This enabled a staged rollout of functionality, shortening the time that was needed for the FLRE to provide value to GNW.

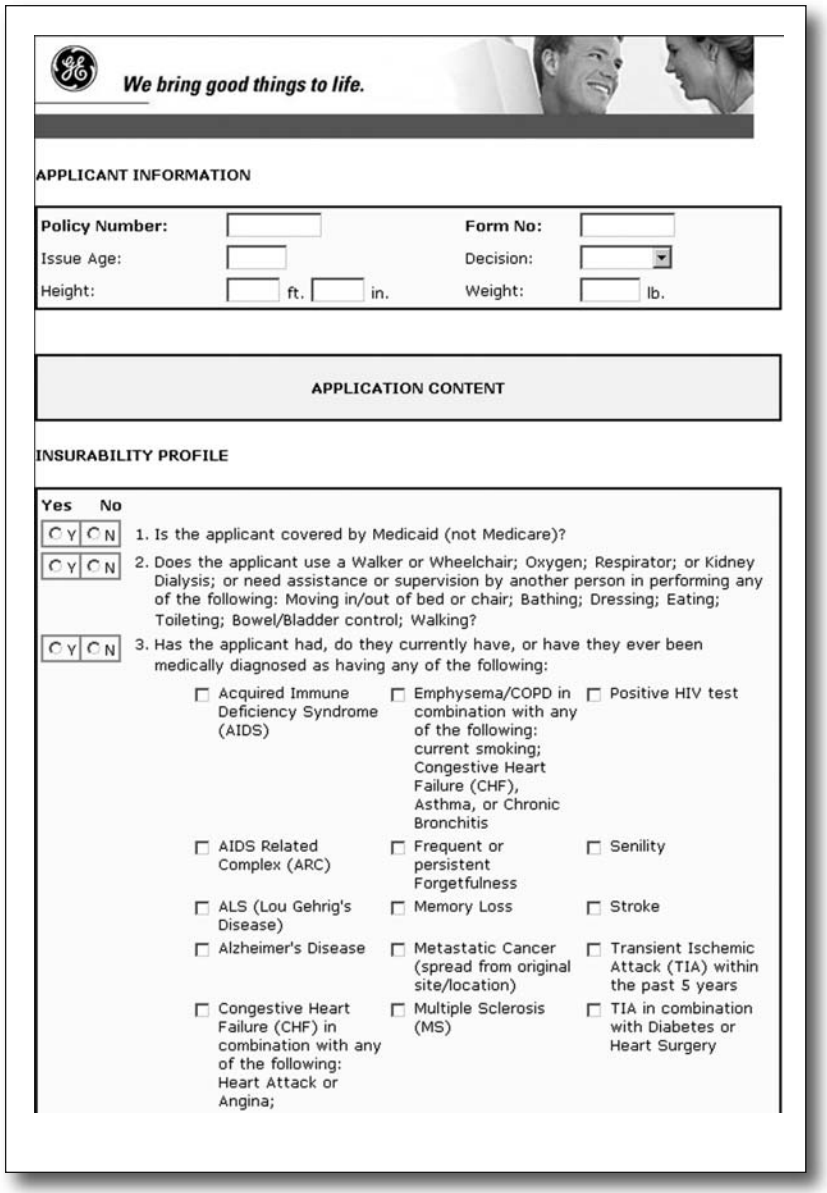


Figure 2. Part One of the APP Summarization Form.

Staged Deployment

Creating an AI system that can solve every instance of a problem can be very difficult. However, an AI system that can solve a subset of the different variations of a problem can still be valuable as long as it can correctly identify which instances it is able to solve. For example, any decision process that is currently being done by humans can benefit from automating the easiest instances of the problem and having people continue to solve the hard instances. The strategy of creating a decision engine to automate a subset of all of the possible cases and determine when this automation is possible can be a useful approach to fielding many types of real-world AI applications.

There are many advantages to this approach, including, for example, (1) the AI system can be fielded more quickly if it handles only a subset of all possible instances; (2) the error rate the AI system will have on the easy cases will be lower than the error rate on all instances; (3) the lower the error rate the more likely the system will be used; (4) deployment issues (integrating with other systems, obtaining user feedback and acceptance) can be handled early in the development. Failure to deal with these issues early can cause an AI system to not be successfully fielded. In addition, early deployment can produce an early return on investment. A system that is being used and providing value is more likely to obtain continued support and interest.

This strategy is not new. The field of software engineering has pointed out many benefits of the cyclical method of software development (Pressman 1987). The cyclical method has a developer create a working system that has only a limited set of features and then add features in future releases. This is particularly useful for unique, novel, or risky projects. Most AI applications would be qualified as unique, novel, or risky (and are often all three).

The AI system should be confident that an instance of the problem is appropriate for it before attempting to solve the instance. If this is not possible, then the system should calculate its confidence in its solution as part of its output (Cheetham and Price 2004). We refer to AI systems that can determine when they are appropriate or provide an estimate of their confidence after they have determined a solution as "confident AI."

The progression of the digital underwriting system through three generations of development and deployment from simple cases to complex is described next.

First Generation

The first generation of the end-to-end system focused on the simplest subset of cases—applications with no medical impairments. The new process begins with a team of medical summarizers digitizing the scanned APPs. The summarizers view the scanned applications online and fill in web-based forms to digitize them. A page from the APP summarization form is shown in figure 2. Next, the digital application is passed through an instance of the FLRE (referred to as the APP-FLRE). The APP-FLRE makes three decisions. First, in what rate class to place the applicant; second, whether or not to order additional information; and third, whether or not to

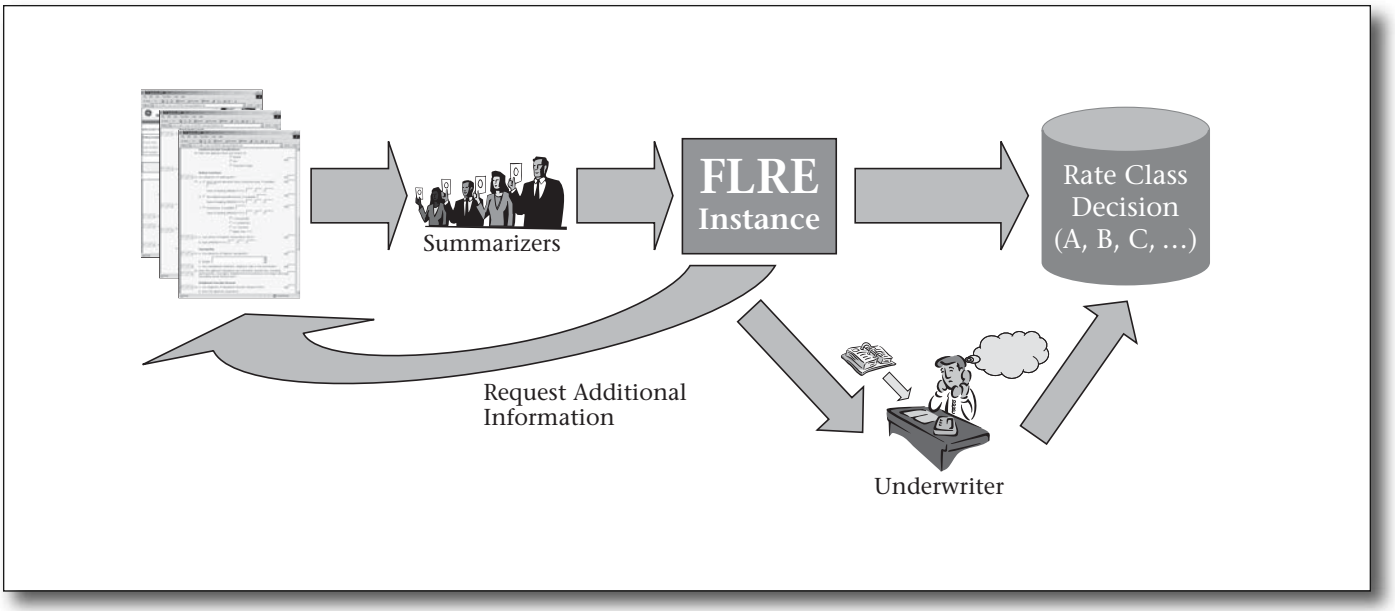


Figure 3. The Automated Underwriting Process.

send the case to a human underwriter for review (that is, reverting to the manual process).

If additional information is requested from the applicant (for example, a PHI or F2F), it is also digitized on arrival. The new content is then passed through separate instances of the FLRE, using different rule sets but making the same three decisions. This new decision process is presented in figure 3.

With multiple decision engines, more than one rate class decision may be made for a single applicant. The lowest rate class (that is, the highest premium) always takes precedence across all of the engines that may be invoked. For example, if an applicant has completed both an APP and a PHI, the lower FLRE decision is used.

If any of the engines decide a case should be sent to a human underwriter for review, that decision will be honored. Cases can be diverted back into the manual process any time an engine is unable to make a definitive decision. If an automated decision is made, a new notification system automatically mails a letter to the applicant with the decision.

Second Generation

The second generation of the system covered two major impairments. Statistics on the frequency of impairments in applications from the past seven years were obtained to drive the specific impairment selection. Figure 4 shows these relative frequencies.

The second generation of engines handled APSs with two of the most common medical impairments: hypertension (HTN) and diabetes mellitus (DM). HTN was chosen because it is the most common impairment seen on applications. DM was chosen because it is also quite common and has one of the highest average claims costs. The coverage of these impairments required new web forms for the summarizers to enter information about the impairments, new rules to determine rate classes from this information, and new rules to determine when applications with these impairments could be automated.

If an APS has been ordered, the medical summarizers review it, determine the applicant's impairments, and then complete the appropriate summarization forms. Separate FLRE instances are invoked as needed.

Third Generation

The third generation focused on three areas: increasing the set of impairments covered, increasing the number of applications that can be automated by adding natural language processing, and assisting the underwriter when an application cannot be fully automated. Two additional impairments were covered by the third generation of the system. Osteoarthritis (OA), the second most frequent impairment, was selected. Osteoporosis (OP) is closely related to OA, so this impairment was also covered.

Natural Language Processing.

After generation two, a significant percentage of the applications containing impairments covered by the rules engines still could not be automated. The primary reason for this was the input from the summarizers occasionally con-

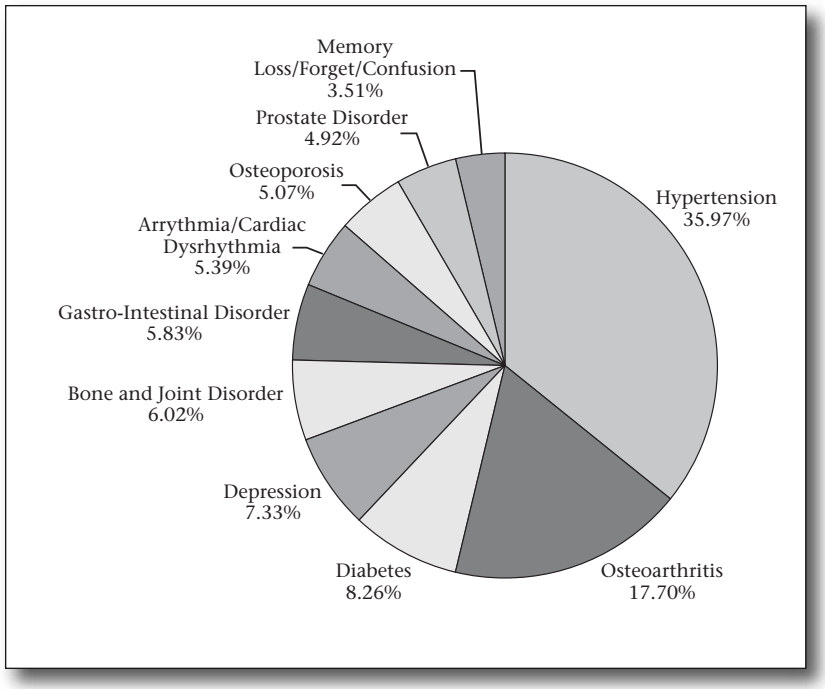


Figure 4. The Relative Frequency of Impairments.

BenignText: BenignPhrase [Separator [BenignPhrase]]*
BenignPhrase: [Noise]* [Benign [Noise]* [Date [Noise]*]

Figure 5. Current Grammar for Benign Text.

Version	False Benign	False Assist	True Benign	True Assist
Basic grammar	1.15	62.54	37.46	98.85
Dates parsed	1.15	62.35	37.65	98.85
Improved lists	0.60	38.08	61.92	99.40
Remove in-phrase characters	0.60	32.56	67.44	99.40
Match longest first	0.83	0.00	100	99.17
? not a separator	0.00	0.00	100	100

Table 1: Natural Language Parser Accuracy.

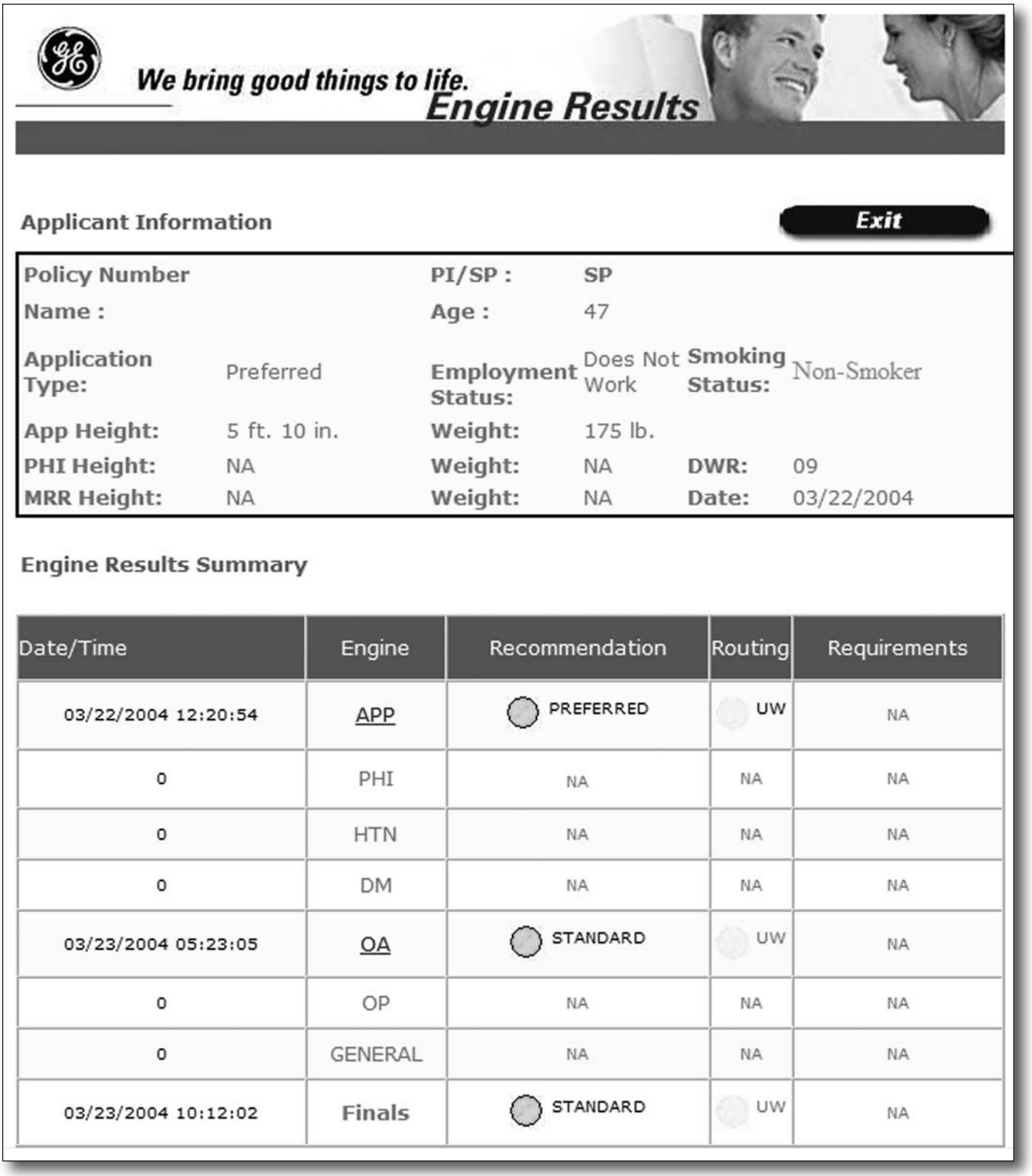
tained free text that required review by an underwriter. Usually this free text did not affect the rate class decision, so if text entries could be interpreted and classified as benign, the level of automation could be increased.

Classifying critical text as benign (that is, false positives) is not acceptable; however, it is acceptable to have errors where benign text is classified as needing review (that is, false negatives). The latter type of errors result in underwriters performing the same tasks they currently do.

A natural language parser (Jurafsky and Martin 2000) was constructed to determine whether the text entered by the summarizers was benign. A grammar was constructed for benign text and lists were created for noise words and in-phrase characters (noise), phrase separators (separator), benign words or synonyms (benign), and dates in various formats (date). The current grammar for benign text is depicted in figure 5.

A training set was used with 160,408 entries, 70.4 percent of which were benign. A list of every unique word in the text was created, and each word was manually classified as benign or not. The evolution of the grammar above is shown in table 1. A basic grammar excluding dates, noise words, and in-phrase characters was developed first. The accuracy of this grammar on the training set is shown in the first row of table 1. The first column represents the percent of text phrases that are not benign but were labeled as benign. These are the most significant classification errors. True benign is the percent of benign phrases that are correctly classified as benign. The larger the true benign, the greater the benefit of the natural language processing feature. A second version of the grammar added parsing multiple date formats, slightly increasing the true benign percentage, as shown in the second row of table 1.

An expanded list of benign terms, which included synonyms and phrases, was then created. This greatly improved the true benign and reduced the false benign rates, as shown in the third row of table 1. To improve the results further, characters such as the dash were treated specially. Next, the parser was modified so that longer phrases had priority over shorter phrases or single words. The true benign rate greatly improved at the expense of a small increase in the false benign rate, as shown in row five of table 1. Finally, question marks were being used as indicators of uncertainty by the summarizers, instead of being at the end of sentences that are questions. Not counting the question mark as a separator produced the final accuracy found in the last row of table 1.



We bring good things to life.

Engine Results

Applicant Information Exit

Policy Number		**PI/SP :**	**SP**		
Name :		**Age :**	47		
Application Type:	Preferred	**Employment Status:**	Does Not Work	**Smoking Status:**	Non-Smoker
App Height:	5 ft. 10 in.	**Weight:**	175 lb.		
PHI Height:	NA	**Weight:**	NA	**DWR:**	09
MRR Height:	NA	**Weight:**	NA	**Date:**	03/22/2004

Engine Results Summary

Date/Time	Engine	Recommendation	Routing	Requirements
03/22/2004 12:20:54	APP	PREFERRED	UW	NA
0	PHI	NA	NA	NA
0	HTN	NA	NA	NA
0	DM	NA	NA	NA
03/23/2004 05:23:05	OA	STANDARD	UW	NA
0	OP	NA	NA	NA
0	GENERAL	NA	NA	NA
03/23/2004 10:12:02	Finals	STANDARD	UW	NA

Figure 6a. Underwriter Assist Screen.

APP

Underwriting Reason	Value	English Rule	Guideline	Source
Prescription_1	NA	Applicant takes a prescription	PDF	pg 2
Speciality_Not_Stated	NA	Speciality not stated	PDF	pg 10
Other_Dr_Reason_Visit_1	NA	Unknown reason for a doctor's visit	PDF	pg 1

Osteoarthritis

Rate Class Reason	Value	English Rule	Guideline	Source
Prescription_Use	NA	Applicant takes a non-narcotic prescription for OA	PDF	pg 7
Joint_Replacement_Discussed	NA	Doctor discussed joint replacement surgery with applicant	PDF	pg 7
COX2_Use	NA	Applicant takes a COX2 inhibitor	PDF	pg 8

Figure 6b. Underwriter Assist Screen (Continued).

After the parser was created, it was tested on a sample population of 36,635 benign and nonbenign phrases. The result from this test set was also 0.00 percent false benign and 100 percent true benign. One reason for these surprisingly good results is the same summarizers were used to produce the training and test data. It is possible the accuracy would decrease if different people created the text phrases.

Some simple non-AI techniques were also used to limit the FLRE cases sent to the underwriter due to free text. This included summarizer training on how and when to enter free text and modifying the entry forms so that common comments could be selected with drop down lists, check boxes, or other nontext-based methods. New rules were created for these new data elements.

Underwriter Assist

The third focus of generation three was to develop a way to help the underwriter when an application could not be placed by the FLREs. This occurred in about 80 percent of the applications. In the first two generations, if an application was sent to an underwriter, he or she had to start on the application from scratch with no visibility into what the FLREs had suggested. For example, if six FLREs had proposed a rate class and one said the underwriter needed to be involved, then the six rate class decisions would all be ignored.

In general, confident AI systems should be able both to automate a subset of the problem instances and to assist the user in solving those instances that cannot be fully automated. Automating a problem clearly saves time for the people who would otherwise need to perform the task. In addition, a confident AI system can be of benefit to users even if it cannot completely automate the task. The system should always provide the most benefit possible for a given instance. The benefit can result from many factors. For example, it can assist by organizing information for the user. It can also assist by identifying what part of the problem it can solve. These items do not need to be inspected by the user, minimizing the time spent on the more mundane tasks. Finally, the system can assist by identifying what keeps the AI system from solving the problem, directing users to those items that require their attention.

For our system, the underwriters' productivity could be improved if the system could propose a rate class for each portion of the application where it was confident in its decision. If an FLRE was not confident, then it should highlight the reason for its lack of confidence. Figure 6 shows a prototype of an underwriter assist screen. The top section has applicant information such as name, age, height, and weight. The next section has a summary of each FLRE result, with one row for each engine. In this example, only the application and the OA-FLRE applied to the applicant, as he or she did not have any other impairment. The engine result summary has five columns: (1)

the date and time the engine was run, (2) the name of the specific engine, (3) the recommended rate class, (4) where to route the application (UW to send to underwriter), and (5) requirements for additional tests needed.

The APP section gives details about the APP-FLRE rules that caused the rate class recommendation and routing. In this example, there was an unknown reason for a doctor visit that needed to be obtained. The underwriter can click the PDF guideline for a complete description of the rule invoked. The original information sent to the summarizer that applied to this rule can be seen by clicking the pages listed in the source column.

The OA-FLRE sent this application to the underwriter because the applicant's doctor discussed joint replacement surgery with the applicant. The underwriter should therefore investigate the severity of the need for surgery, which would significantly affect the applicant's rate class. This interface provides the underwriter with the ability to get an immediate assessment of the applicant and focus attention on the problem areas instead of having to review the entire application.

Use of AI Technology

Fuzzy logic rules are used to encode underwriting standards. Fuzzy logic is a superset of conventional Boolean (true/false or 1/0) logic, allowing truth values to be equal to any real number in the interval [0,1], with intermediate values denoting a "partial degree of satisfaction" of some statement or condition (Zadeh 1965). Each rule represents fuzzy constraints at the boundaries between different rate classes for each input, such as cholesterol, blood pressure, or body-mass index.

Evolutionary algorithms are also used in the new system, to optimize the numerical parameters in the fuzzy logic rules. The use of both fuzzy logic and evolutionary algorithms is described next. As discussed previously, natural language processing techniques were also used to increase the capacity of the automated system.

Fuzzy Logic Rules Engine

The fuzzy logic rules engine was designed to handle discrete classification problems in which the decision categories form an ordered set (Bonissone, Subbu, and Aggour 2002). The FLRE was implemented within a reusable, optimizable architecture for decision systems (ROADS), a generic framework designed to facilitate the implementation of intelligent decision engines (Aggour and Pavese 2003). The engine makes decisions through a three-step process. First, rule evaluation through fuzzy membership functions, second, aggregation evaluation and threshold application, and third, assignment of final decision (defuzzification).

A separate membership function is defined for each input for each rate class, to specify distinct cutoffs for each. Cutoffs were initially derived from knowledge engineering sessions with expert underwriters, and later optimized using an evolutionary algorithm.

When the FLRE makes a decision, the input data is passed through each of the fuzzy membership functions and scores are generated. After the rule scores have been generated, an aggregation is performed for each rate class. The scores are passed to each aggregation operation, which creates a single fuzzy score for each rate class in [0,1].

For each of these rate class scores, a pass/fail test is performed using a threshold value. Each rate class may specify different criteria for whether the tests pass or not. The rate classes are tested in the order of best to worst. The first rate class that passes all criteria becomes the final decision of the engine.

The FLRE is extremely flexible. Different membership functions can be defined for both continuous and discrete inputs. For continuous inputs, membership functions such as step (Boolean), trapezoidal, and generalized bell can be defined. For discrete inputs (such as binary), a fuzzy score can be associated with each possible value. Various functions can be used for the aggregation, including min, max, and average operations.

Figure 7 shows a representation of three rules for one rate class, referred to as Rate Class A. For this example, the membership functions are trapezoidal and the aggregation is a min operation. The final step is for the engine to determine whether the score of 0.8 falls within the threshold for rate class A. If it does, the applicant is assigned to this rate class.

Evolutionary Algorithm Optimization

The FLRE uses an evolutionary algorithm (EA) provided within ROADS for automated parameter tuning. Each chromosome in the EA contains a vector of tunable FLRE parameters. These elements typically represent membership function parameters (core and support values, for example), aggregation parameters, and threshold values. It is up to the system designer to specify what parameters to tune and what values to keep static. Any subset of the parameters may be tuned at the discretion of the user—the ROADS EA generates the chromosome structure based on values set in an XML configuration file loaded at run time.

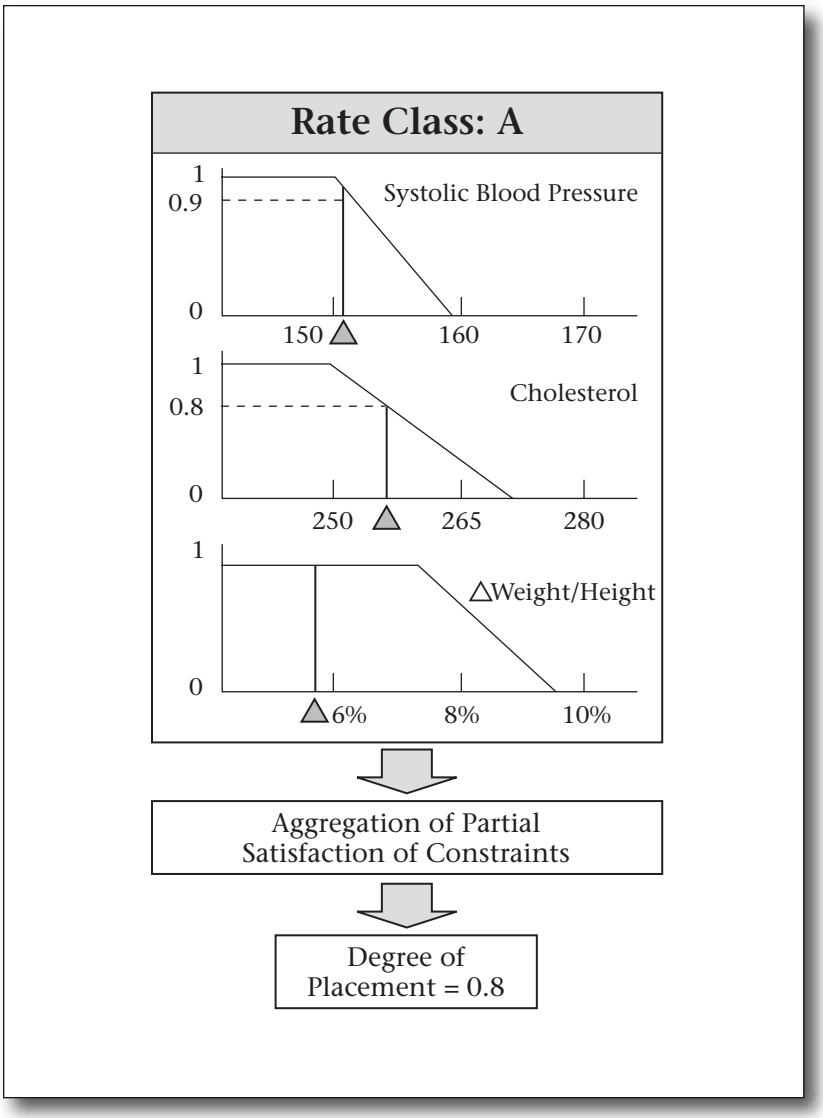


Figure 7. Fuzzy Rule Evaluation.

Since a chromosome defines a complete configuration of the FLRE, at each generation of the EA every chromosome in the current population initializes a separate instance of the FLRE, as shown in figure 8. The engine is run against a set of test cases, each of which has a benchmark decision associated with it. With this information, we can create a confusion matrix *M* that compares the FLRE decisions versus the benchmark decisions. In this matrix, the ordered rate classes (left to right, and top to bottom) correspond to increasing risk categories. The rows of this matrix correspond to certified case decisions as determined by an expert human underwriter, and the columns of this matrix correspond to the FLRE decisions for the cases in the certified case database.

The element $M(i,j)$ represents the frequency of applications belonging to rate class *i*, which were placed by the system in rate class *j*. Elements on the main diagonal, $M(i,i)$, show frequencies of correct classifications. Elements above the main diagonal show frequencies of misclassifications in which the FLRE was too strict and assigned the applications to a higher than needed rate class. This situation leads to noncompetitive pricing and a potential decrease in the volume of placed policies. Elements below the main diagonal represent the opposite situation, in which the FLRE was too lenient and assigned applications to lower than needed rate classes. This situation leads to the acceptance of excessive risk without proper premiums. We want to use a fitness function that provides a balance between price competitiveness and risk avoidance, considering the asymmetry of their costs.

From actuarial studies we derived a cost matrix *P*, such that element $P(i,j)$ represents the cost of misclassifying rate class *i* as *j*. This cost is the *loss of net present value (NPV)*[1] caused by this error. By multiplying element-wise the confusion matrix *M* with *P*, we obtain the cumulative *expected loss of net present value* for the classifier instantiated by each chromosome and evaluated over the training set using a leave-one-out technique. This fitness function is used to rank the chromosomes in the population, determining how likely each is to be selected for crossover and mutation, as illustrated in figure 8. This computation, which captures the tradeoff between price competitiveness and risk avoidance, is fully described by Bonissone, Subbu, and Aggour (2002) and Yan and Bonissone (2006).

System Description

The automated underwriting system has a number of components, each executing on Microsoft Windows 2000 operating systems. As the summarizers digitize applications through their web interface, the digitized information is stored in an Oracle database for further processing. Every 15 minutes, a process is initiated that queries this database for any new cases. If the summarizers have entered a new case, it is extracted from the database, the appropriate FLRE is instantiated and the case is evaluated. The output is then stored in the same Oracle database.

The FLRE was implemented entirely in Java 1.3.1 so that it can run in both UNIX and Microsoft-based environments without requiring recoding. Once initialized, the engine takes fractions of a second to execute each case. The engine was designed and developed entirely in-

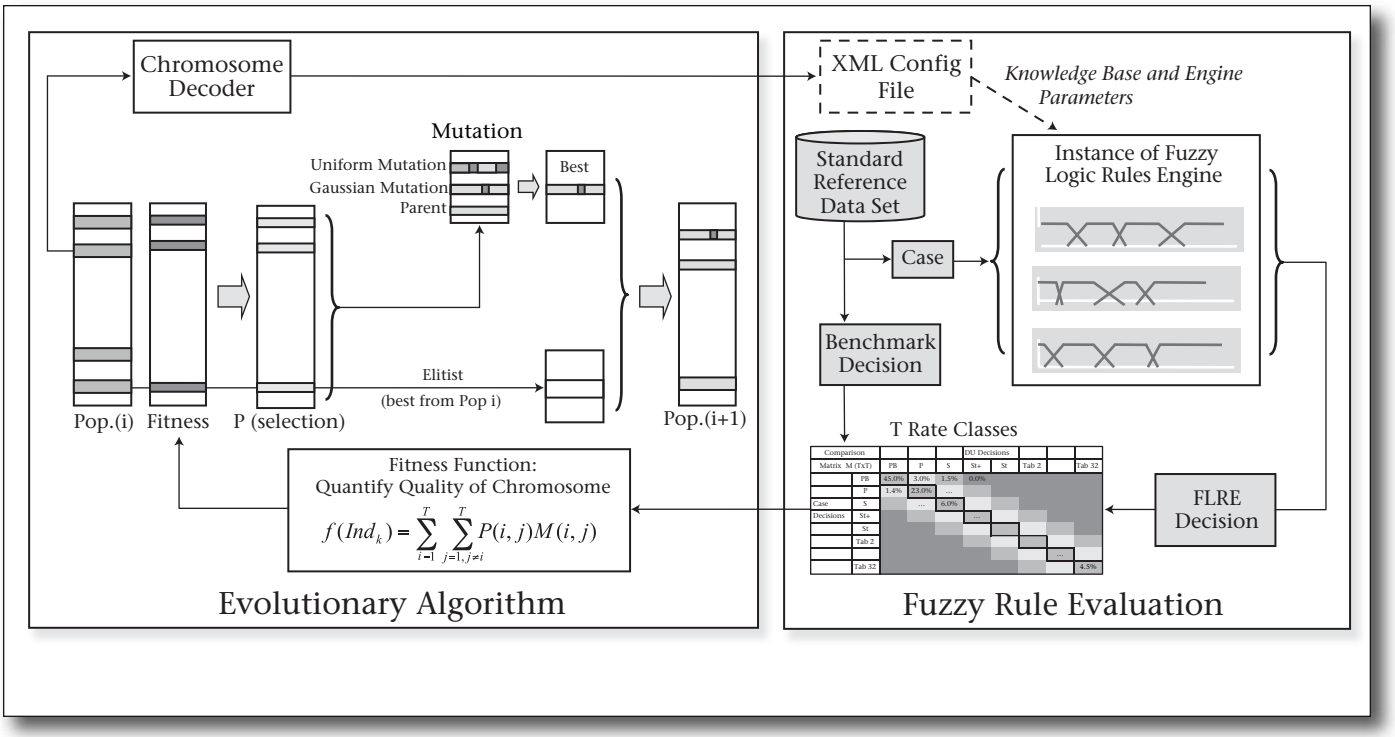


Figure 8. FLRE Optimization Using an Evolutionary Algorithm.

house. Third-party tools were reviewed, but at the time none had the desired flexibility to represent underwriter knowledge in fuzzy rules that could be aggregated and tested against a threshold.

While multiple rules exist per rate class, repeated rule chaining was not allowed out of concerns for maintainability and readability. If a rule's result is an input to a second rule, then the output of the second rule cannot be used as input to any other rule.

Application Use and Payoff

Generation one was deployed in December 2002. It automated 12 percent of the LTC underwriting volume. Generation two was deployed in May 2004, increasing the percentage of automated applications to 19 percent. All (100 percent) of the applications are now digitized and sent to the APP-FLRE. In 2004, the average weekly volume sent to the APP-FLRE was 3,500 applications. Accuracy on the automated applications is nearly a hundred percent. Generation three has been coded, is currently being tested, and is scheduled to go into production shortly.

Before this system, 14 percent of all PHIs ordered were never used. The underwriters are now prevented from ordering PHIs, and the engine orders only what is needed. Assuming the underwriters had continued ordering at the same error level, the savings on this aspect alone calculate to approximately $500,000 per year.

Automating this process had a number of other benefits, including improving decision consistency and significantly reducing the number of incorrect decisions. Reducing defects allows GNW to remain price competitive while effectively managing risk. And with an efficient, automated process handling a portion of the case volume, the capacity of the underwriting organization has increased.

In May of 2004, Genworth Financial was spun off from the General Electric Company. At the time of the IPO, stock analysts cited this advanced technology as one of the key advantages GNW has over its competitors.

Application Development and Deployment

Much of the work in deploying an AI application goes beyond the AI portion of the system development. The four major portions of this project were collecting, verifying, and standardizing the knowledge; digitizing the inputs and outputs and integrating with existing systems

and processes; creating the AI system; and creating tools to monitor and maintain the system.

Each of these portions required a significant effort to be completed successfully. The FLRE was designed and developed by four engineers over a period of six months. The underwriter guidelines were collected initially from the underwriter manual and then reviewed and updated with a committee of two underwriters and GNW's medical expert, requiring roughly three months of effort. The spiral development model was followed for the design and development of the FLRE and the implementation of the underwriter rules in the engine. The summarizer form creation and testing required about two months of effort from one engineer, two underwriters, and three representatives of the summarization team. The prototyping development model was followed for the implementation of the summarizer forms, as they required numerous iterations.

Data collection and validation took approximately four months for two of GNW's IT professionals. By far the most difficult step in the process was data collection and cleaning. Historical data was readily available to validate the decision engine and test the end-to-end process, but the quality of that data was less than ideal. Some cases were incomplete, and others did not have associated final decisions. A key takeaway for the team was never to underestimate the amount of time and effort required for handling data issues.

A diverse group invested a significant amount of time and effort to design, implement, and deploy the complete end-to-end system. This included AI experts, IT professionals, statisticians, underwriters, medical experts, and summarizers. While the use of AI technology was critical to the success of the project, it was only a component of the new system. Over $1 million was spent over the course of a year and a half to develop and test the end-to-end system. An additional four months was spent deploying the system. Deployment required multiple steps. A new process was developed to allow medical summarizers to digitize the paper applications. A Microsoft Windows 2000 machine was configured to run a web server hosting the summarizer's web interfaces, which also required the integration of a new Oracle database to store the digital underwriter's data records. A backup database was also deployed to act as a failover, in case the primary database failed. These two databases were linked so that any changes in the primary database were reflected in the backup.

A new program was written to query the database automatically for new cases at a timed interval and to instantiate appropriate instances of the FLRE as required. This program stores the results from the FLRE in the same Oracle database. Before this system could be used in production, a new process was also defined to generate notification letters automatically to the applicants when final decisions were made.

During the four-month deployment period, the entire new process was closely monitored with particular attention paid to auditing the engine's decisions. Initially, 100 percent of the engine's decisions were reviewed. After roughly a month's worth of correct decisions were produced and the user's confidence in the automated decisions grew, the auditing was reduced to 25 percent of the cases. This continued for another month, followed by a drop to 10 percent and eventually 5 percent of cases being audited, today's approximate steady state.

The following process was followed for the development and deployment of each generation: (1) knowledge acquisition from underwriter manual and review of guidelines, (2) transform guidelines into rules, (3) review rules with experts and users, (4) code rules and summarizer entry forms, (5) test on 100 examples, (6) review results with experts (7) update rules and forms, (8) work with IT to install new rules and forms, (9) test on 400 more examples, (10) update rules and forms, (11) write training material, (12) release to pilot group, (13) review results of pilot, (14) update rules and forms, (15) finalize training material, (16) release to production, (17) sample 5 percent of volume processed, and (18) monthly review of sample.

This process ensures that rules are never placed into production without a thorough evaluation, and after release they are reviewed to ensure they are performing as expected.

Sixteen patents have been submitted to the U.S. Patent and Trademark office, covering many aspects of the automated underwriting process. These include "System for Summarizing Information for Insurance Underwriting Suitable for Use by an Automated System," "System for Rule-Based Insurance Underwriting Suitable for Use by an Automated System," "System for Optimization of Insurance Underwriting Suitable for Use by an Automated System," and "System for Determining a Confidence Factor for Insurance Underwriting Suitable for Use by an Automated System."

Monitoring and Maintenance

A serious challenge to the successful deployment of intelligent systems is their ability to remain valid and accurate over time, while

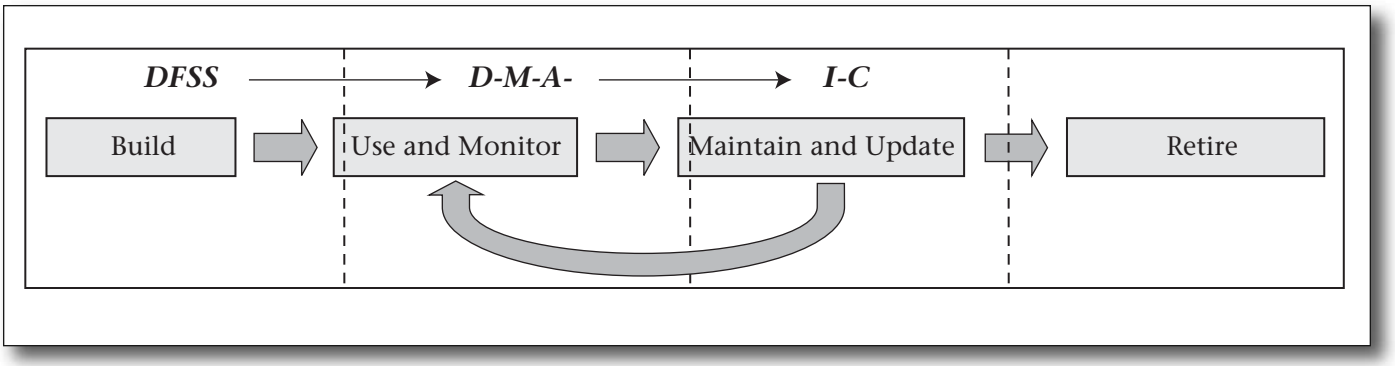


Figure 9. Lifecycle of an Automated Decision Engine.

compensating for drifts and accounting for contextual changes that might otherwise render their knowledge bases stale or obsolete. This issue has been a constant concern in deploying AI expert systems and continues to be a critical issue in deploying knowledge-based classifiers. The maintenance of the classifier is essential to its long-term usefulness since, over time, the configuration of the engine may become suboptimal. Therefore, before deploying a model in a production environment we must address the model's complete life cycle, from its design and implementation to its validation, production testing, use, monitoring and maintenance. By maintenance we mean all of the steps required to keep the model vital (that is, nonobsolete) and adaptable.

Two reasons justify our emphasis on maintenance. First, over the life of the model, maintenance costs are the most expensive (as software maintenance is the most expensive component of the software life cycle). Second, when dealing with mission-critical software we need to guarantee continuous operations or at least fast recovery from system failures to avoid lost revenues and other business costs.

Taken from Patterson, Bonissone, and Pavese (2005), figure 9 shows a model's lifecycle within a six sigma quality framework. The FLRE was developed using a design for six sigma (DFSS) approach including optimization based on evolutionary algorithms. The model use, monitoring, maintenance, and updates follow a design, measure, analyze, improve, and control (DMAIC) approach that shares the same evolutionary algorithm for the improve and control phases.

While the system is used in production, we need to monitor it in real-time to generate requirements for the next regular update and to identify situations in which the engine may require immediate updating before those situations turn into significant problems. Finally, we need to update and maintain the model to incorporate the change requirements generated during the monitoring phase.

Real-Time Monitoring

Before deployment, we performed a failure mode and effects analysis (FMEA) to identify the possible ways in which the system could fail and the consequences of those failures (Patterson, Bonissone, and Pavese 2005). These modes were prioritized to identify the critical few factors that were most important for monitoring postdeployment. This exercise was valuable because once a new AI system has been deployed in production much of the real work begins. This includes the traditional maintenance performed by IT, including managing and maintaining the web servers and databases used. While necessary, this is not sufficient. Regular underwriter auditing of the FLRE is also critical to ensure that the engine is correctly classifying policies over time. This includes auditing both the coverage and accuracy of the engine. A decrease in coverage would result in a reduction in productivity of the underwriting team, since decisions that cannot be automated must be resolved manually. The engine's accuracy may be affected if there is a shift in the applicant population (such as age distribution), for example.

As previously mentioned, the underwriters regularly audit roughly 5 percent of the engine's decisions. When we deployed the FLRE for term life insurance underwriting, we created an offline quality assurance (QA) process to support the auditing process. The QA process consisted of four independent classifiers based on neural networks, multivariate regression splines, support vector machines, and random forests. We leveraged the diversity of these components and fused them to create a highly reliable rate class decision, to test and monitor the production FLRE that performed

the online rate classification. At periodic intervals, we used this QA process to review the decisions made by the FLRE over that period of time. In addition, this fusion process identified the best cases to be used for tuning the production engine, as well as controversial or unusual cases to be audited or reviewed by human underwriters (Bonissone, Eklund, and Goebel 2005, Bonissone 2006).

Not only is it important to monitor the output of the engine, but it is also valuable to monitor its inputs, to verify the accuracy of the data being sent to the engine, and to analyze the distribution of cases being processed. This real-time monitoring allows us to verify that there are no repeating defects in the summarization process and to understand the population of cases being processed by the engine. Significant changes in the population distribution would signal a potential need for updating the rules or even creating new or modifying the existing rate classes. To allow for this level of detailed analysis, it is important that all data throughout the decision process be stored and monitored. This includes the data entered into the engine, the individual rules that are fired for each instance of the FLRE, and each FLRE decision.

Maintenance

The system is maintained in three ways. First, major updates are made with every generation deployed. Second, minor updates are deployed between major updates. Finally, parameter tuning can be performed with the evolutionary algorithm.

LTC underwriting rules do not change often. Consequently, the majority of changes have been included with the generation releases. If a change is made to the underwriting guidelines, the maintenance team can also deploy changes to the FLRE between generations. However, the primary reason for changing the underwriting guidelines has been clarifications needed to create rules from the guidelines in the first place. These clarifications in the guidelines are a side benefit of constructing the FLRE. Between-generation changes go through the thorough testing process described in the application development and deployment section.

Perhaps one of the most interesting aspects of the FLRE design and maintenance was its error-cost–based derivation. The FLRE's parameters and decision thresholds were first initialized by elicitation from expert underwriters and then tuned using a mutation-based evolutionary algorithm wrapped around the classifier to achieve a specific trade-off between accuracy and coverage. The fitness function selectively penalized different degrees of misclassification according to their costs, expressed in net present value, and served as a forcing function to drive correct classifications and minimize the costs of misclassifications.

Final Thoughts

The application of AI in this knowledge domain has reinforced a key learning from previous applications. Namely, recognizing the importance of breaking the ultimate goal into a series of intermediate steps so that users can build the necessary "comfort level" is critical for success. New users of a technology want to be able to slowly increase their comfort level with the technology and learn how it works and understand how it makes decisions. If the technology is to help them, they want to see it applied to simple cases and see in simple language how it works. They want to verify that the decisions agree with theirs and see how they were arrived at. Most industrial users are not familiar with AI technology, so they can't be expected to go from algebra to advanced calculus in one step (to use an analogy)!

The process described for implementing the generations toward the final goal was critical from the whole team's perspective. It also had an additional benefit: as the decision complexity increased, new capabilities of the FLRE had to be added, and a deployment process created. This provided a natural way for the transition of knowledge from the research and development environment to the business environment. After the project was complete, there needed to be a familiarity and process for changing existing and adding new capabilities to the automated underwriting process resident within the business. The multigeneration approach allowed for the responsible business team to become experts in how to modify and enhance their automated underwriting capabilities.

Another important point in deploying such applications and in choosing opportunities for creating new applications is the necessity to deliver applications that produce robust decisions. Situations in which a small change in a decision variable can cause a large change in the decision are considered to be nonrobust (or brittle). The system has to be rigorously tested to ensure this behavior does not occur. The use of fuzzy rules in the underwriting application was an effective way to mitigate the potential for brittleness in the automated underwriting system. In fact, the underwriters felt that strict Boolean rules produced decisions that they

were not comfortable with, largely because of this lack of "robust" behavior. As applications with increasing decision complexity are tackled, this robustness issue will likely be of increasing importance. Hence, it is probably still a good topic for additional research, particularly with stochastic variables.

Conclusions

The automation of the underwriting of insurance applications has been a success. The artificial intelligence components (fuzzy logic rules engine, evolutionary algorithm, and natural language processing) enabled this success, but they were just one portion of the changes needed. This project required updating the underwriting guidelines, changing the underwriting process, switching the application process from paper-based to digital, adding personnel to digitize the summaries, and automating the creation of notification letters. The AI techniques were useful because they were a part of a larger end-to-end system.

We established a reliable, repeatable process to design and maintain the FLREs. In our approach we designed the classifiers around a set of standard reference decisions (SRD), which embodied the results of the ideal behavior that we wanted to achieve during development and that we wanted to track during production use.

During the life of the classifier we might need to change the underwriting rules. These modifications could be driven by new government regulations, changes among data suppliers, new medical findings, and so on. These rule changes are used to update the SRD. The updated SRD represents the new target that we want our classifier to approximate. At this point, we can use the same EA-based optimization tools employed during the initial tuning to find a parametric configuration that structures the FLRE to better approximate the new SRD.

In the future, FLREs for other impairments are planned in the order of the value of their addition, where the value is the cost of the current manual process minus the cost of creating, maintaining, and utilizing the forms and rule sets. Another group in GNW is creating a web-based customer self-service application that will use the FLREs to give immediate rate quotes when all of the required data is available.

Acknowledgements

We would like to thank Helena Goldfarb and Barry Hathaway at GE, and Amy Chambers, Dilip Chemburkar, Evely Euripides, David Gorman, Scott Hansen, JoAnn Hurley-Tuel, Bruce Margolis, Brian McCutcheon, Marc Pavese, Craig Robinson, Sylvia Soden, Rick Storms, Tammy Wood, and Jerry Zeng at Genworth for their contributions to this project.

Note

1. We define ***NPV = PR – PC***, where *PR* is the *present revenue* and *PC* is *present cost*. Cells *above* the main diagonal (overestimated risks) have lower *PR* (since higher, noncompetitive premiums will cause more policy lapses) and the same *PC* across the row (since they all have the same risk). Cells *below* the main diagonal (underestimated risks) will have higher *PC* (since higher risks will cause higher likelihood of claims), while *PR* will not be large enough to compensate for the increased risk. This explains the lack of symmetry in the cost matrix ***P***.

References

Aggour, K. S., and Pavese, M. 2003. ROADS: A Reusable, Optimizable Architecture for Decision Systems. In *Proceedings of the 15th International Conference on Software Engineering and Knowledge Engineering*, 297–305. Lecture Notes in Computer Science. Berlin: Springer-Verlag.

Bonissone, P. P. 2006. Automating the Quality Assurance of an On-line Knowledge-Based Classifier by Fusing Multiple Off-line Classifiers. In *Modern Information Processing: From Theory to Applications,* ed. B. Bouchon-Meunier, G. Coletti, and R. R. Yager, 147–157. Amsterdam: Elsevier.

Bonissone, P. P.; Eklund, N.; and Goebel, K., 2005. Using an Ensemble of Classifiers to Audit a Production Classifier. In *Proceedings of the 6th International Workshop on Multiple Classifier Systems*, 376–386. Lecture Notes in Computer Science 3541. Berlin: Springer-Verlag.

Bonissone, P. P.; Subbu, R.; and Aggour, K. S. 2002. Evolutionary Optimization of Fuzzy Decision Systems for Automated Insurance Underwriting. In *Proceedings of the 2002 IEEE International Conference on Fuzzy Systems*, 2, 1003–1008. New York: Institute of Electrical and Electronic Engineers.

Cheetham, W., and Price, J. 2004. Measures of Solution Accuracy in Case-Based Reasoning Systems. In *Proceedings of the Seventh European Conference on Case-Based Reasoning*. Lecture Notes in Computer Science 3155. Berlin: Springer-Verlag.

Chisholm, M., 2004. *How to Build a Business Rules Engine*. San Francisco: Morgan Kaufmann.

Jang, R.; Sun, C.; and Mizutani, E. 1997. *Neuro-fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence*. Upper Saddle River, NJ: Prentice Hall.

Jurafsky, D., and Martin, J. 2000. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Upper Saddle River, NJ: Prentice Hall.

Patterson, A.; Bonissone, P. P.; and Pavese, M. 2005.

Call for Participation:

AAAI-07 Student Abstract and Poster Program

AAAI-07 invites submissions to the student abstract and poster program, to be held July 22–26, 2007 in Vancouver, British Columbia. The goal of this program is to provide a forum in which students can present and discuss their work during its early stages, meet some of their peers who have related interests, and introduce themselves to more senior members of the field. The program is open to all pre-Ph.D students. Nonstudent advisors or collaborators should be acknowledged appropriately, as coauthors or otherwise. However, students are requested to honor the spirit of the program by submitting only work for which they are primary investigators.

Submissions and Dates

Electronic submission in PDF format is required. Students should submit an abstract describing their research no later than January 25, 2007. Abstracts must be no longer than 2 pages including references, and formatted in AAAI two-column, camera-ready style. Instructions about how to submit abstracts will be available at the AAAI conference web site (www. aaai.org/Conferences/AAAI/aaai07.php) after October 1, 2006. Papers exceeding the specified length and formatting requirements are subject to rejection without review.

The abstract must include the following: title; the primary author's full name, affiliation, postal address, phone number, URL (if available), and e-mail address; all coauthors' full names and affiliations; text; and any figures, tables, or diagrams. The abstract should also contain a URL of a location where reviewers can access complementary material about the student's research. The URL is critical to reviewers because of the brevity of the hard-copy submission.

Notification of acceptance or rejection of submitted abstracts will be mailed to the author by March 23, 2007. Camera-ready copy of accepted abstracts will be due by April 10, 2007.

Submissions to Other Conferences

Students are free to submit abstracts for work reported in a regular paper submitted to AAAI-07 or another conference, but not for work that has already been published. Abstracts will be accepted or rejected for the student session regardless of the outcomes of related paper submissions.

Publication

Accepted abstracts will be allocated two (2) pages in the conference proceedings. Students will be required to transfer copyright of the abstract to AAAI.

Poster Session

Accepted abstracts will be allocated presentation time and space in the student poster display area at the conference. Student authors of accepted abstracts must agree to prepare a poster representing the work described in their abstracts and to be available to discuss their work with visitors during their allocated time in the student poster display area.

Student Abstract Inquiries

Registration and call clarification inquiries may be sent to:

AAAI-07 Student Abstracts
American Association for Artificial Intelligence
445 Burgess Drive
Menlo Park, CA 94025-3442 USA
aaai07@aaai.org

All other inquiries and suggestions should be directed to:

Mehran Sahami, Student Abstract & Poster Cochair
Google Inc. (sahami@google.com)

Kiri Wagstaff, Student Abstract & Poster Cochair
Jet Propulsion Laboratory (kiri.wagstaff@jpl.nasa.gov)

Matt Gaston, Student Abstract & Poster Cochair
University of Maryland Baltimore County (mgasto1@cs.umbc.edu)

Six Sigma Quality Applied throughout the Lifecycle of an Automated Decision System, *International Journal of Quality and Reliability* 21(3): 275–292.

Pressman, R. 1987. *Software Engineering: A Practitioner's Approach*. New York: McGraw-Hill.

Yan, W., and Bonissone, P. P. 2006. Designing a Neural Network Decision System for Automated Insurance Underwriting. In *Proceedings of the IEEE Joint International Conference on Neural Networks*. New York: Institute of Electric and Electronic Engineers.

Zadeh, L., 1965. Fuzzy Sets. *Information and Control* 8(3): 338–353.

Kareem S. Aggour is a computer engineer at General Electric's Global Research Center in upstate New York, where he focuses on the design and development of AI systems for industrial applications. Most recently, he has performed research in the fields of intelligent information retrieval and extraction. He earned B.S. degrees in electrical engineering and computer science from the University of Maryland, College Park (1998) and an M.S. in computer engineering from Rensselaer Polytechnic Institute (2001). He is currently pursuing a Ph.D. at Rensselaer part time. His e-mail address is aggour@ research.ge.com.

Piero P. Bonissone is a senior project leader and Coolidge Fellow at GE Global Research. He has been a pioneer in the field of fuzzy logic and approximate reasoning applications. He is a Fellow of AAAI, IEEE, and IFSA. In 2002 he was president of the IEEE Computational Intelligence Society. He has been an adjunct professor for the past 24 years at Rensselaer Polytechnic Institute in Troy, NY. He is the author or coauthor of more than 130 technical publications (including four books) and holds 33 patents. His e-mail address is bonissone@ research.ge.com.

William E. Cheetham is a senior researcher in the artificial intelligence laboratory at GE Global Research, where he has worked since 1985. His job is to invent or improve knowledge-based products and services. This often involves the application of artificial intelligence and related techniques. He has led the development of more than a dozen intelligent systems that are in use throughout the General Electric Company. He has been an adjunct professor at Rensselaer Polytechnic Institute since 1998, where he teaches the class Applied Intelligent Reasoning Systems. His e-mail address is cheetham@research.ge.com.

Richard P. Messmer is a senior project leader and Coolidge Fellow at GE Global Research. His research interests over the years have been in quantum and statistical physics, material science, decision science, quantitative finance, and risk analytics. He has been a visiting professor at Caltech and was an adjunct professor of physics for 18 years at the University of Pennsylvania in Philadelphia, PA. He is the author or coauthor of more than 170 technical publications. His e-mail address is messmer@research.ge.com.