AI Magazine Volume 20 Number 1 (1999) (© AAAI)

The NASD Regulation Advanced-Detection System (ADS)

J. Dale Kirkland, Ted E. Senator, James J. Hayden, Tomasz Dybala, Henry G. Goldberg, and Ping Shyr

■ The National Association of Securities Dealers, Inc., regulation advanced-detection system (ADS) monitors trades and quotations in The Nasdaq Stock Market to identify patterns and practices of behavior of potential regulatory interest. ADS has been in operational use at NASD Regulation since the summer of 1997 by several groups of analysts, processing approximately 2 million transactions a day, generating over 10,000 breaks. More important, it has greatly expanded surveillance coverage to new areas of the market and to many new types of behavior of regulatory concern. ADS combines detection and discovery components in a single system that supports multiple regulatory domains and shares the same market data. ADS makes use of a variety of AI techniques, including visualization, pattern recognition, and data mining, in support of the activities of regulatory analysis, alert and pattern detection, and knowledge discovery.

The National Association of Securities Dealers, Inc. (NASD), has been regulating the securities industry since its founding in 1939. Regulation of securities markets and firms is undertaken by its NASD Regulation, Inc., subsidiary. Our mission of investor protection includes monitoring trading and quotation activities on The Nasdaq Stock Market, the Over the Counter (OTC) Market, and the Third Market to identify and correct any potential violative activities by more than 5500 member firms. Central to this job is the advanced-detection system (ADS).

We have been using ADS since the summer of 1997 to provide analysts in the Market Regulation Department with significant leads to potential patterns or practices of regulatory concern, or *breaks*. ADS generates these breaks by integrating and then reviewing all quotation and trade records, almost 2 million on a typical day, for patterns that indicate the occurrence of targeted scenarios.

Since beginning production operations, the system has detected over 10,000 breaks, of which more than 10 percent have merited follow-up actions of various types, a threefold increase in effectiveness compared to previous techniques. More important, it has greatly expanded our surveillance coverage to new areas of the market and to many new types of behavior of regulatory concern.

ADS relies on a rule pattern matcher and a time-sequence pattern matcher. Two- and three-dimensional visualizations allow analysts to see the market context of breaks and temporal relationships of events in large amounts of data. Data-mining tools permit discovery of new patterns of potential regulatory interest.

ADS is evolving continuously to remain current with changes in market behavior, increase its effectiveness, add other features, incorporate additional market data, cover additional types of potential violation, and keep up with improvements in market structure.

Task Description

Nasdaq is a screen-based dealer market consisting of competing market makers who risk their own capital to provide liquidity. A *market maker* provides quotations for issues in which he/she makes a market. A *quotation* consists of both a *price* and a *size* (number of shares) at which he/she is willing to buy or sell, respectively, a particular security, or *issue*. The price at which he/she is willing to buy is known as a *bid* and at which he/she is willing to sell as an *ask* or *offer*. The highest bid and lowest ask at a particular time are known as the *inside quote*. Quotations are available to other market makers, other brokers, and investors through the

distributed computing system that forms the heart of The Nasdaq Stock Market. Trades are executed between market makers (acting as principals for their own account or as agents for customers) and dealers (acting as agents for customers) using one of several automated systems. Trades are reported, shortly after they occur, to a common system, resulting in the well-known stock ticker tape.

Nasdaq currently has more than 5500 issues on the Nasdaq National Market and The Nasdaq SmallCap Market. There are an average of about 10 market makers an issue. A typical day consists of over 900,00 quote updates, 400,000 inside quote updates, and 800,000 trade reports on both tiers of The Nasdaq Stock Market and the OTC market.

Individual trades or quotes can often be justified before disciplinary committees. Broad patterns and practices, however, cannot. A key goal of ADS is to detect patterns and practices of violative activity. Another goal of ADS is to raise the level of surveillance from issue-based to firm-based patterns and practices.

A key goal of ADS is to detect patterns and practices of violative activity. Another goal of ADS is to raise the level of surveillance from issue-based to firm-based patterns and practices.

Our data problem is, in part, statistical. Statistical techniques are effective at identifying outliers. However, outliers often occur in the context of unusual market activity, and potential violations occur during normal market conditions. Even after a potential concern is identified, an analyst needs to review large amounts of market information to determine if there is a potential explanation for the apparent violation. Prior to ADS, analysts reviewed this information in tabular formats, from which it was difficult to discern relationships. We needed the right mix of statistics, data classification, data visualization, and pattern recognition on a huge database of activity structured in time sequence.

ADS currently covers three areas, or domains, of potential violation, each with its own user team having a distinct set of business procedures and needing unique data, knowledge, and tools to perform its function. However, because of the large overlap between necessary and available data and tools, a single system was customized to meet the needs of each team. This approach is allowing us to cost effectively extend ADS to additional domains as requirements demand and resources permit.

Late-Trade Reporting

To provide timely and accurate information to the marketplace, trades must be reported within 90 seconds of execution. However, for various reasons, trades are sometimes not reported within this window. The Market Regulation Department is responsible for surveillance of all late reported trades. It determines whether the late-trade reporting is abusive or not, whether there is a pattern or practice of late-trade reporting, and what regulatory action to initiate against reporting firms. An example that could initiate regulatory action is a trader delaying the reporting of a large customer trade so that the customer doesn't see that there was another trade at a better price at the same time. A market maker has a much higher incidence than his/her peer firms of reporting trades late.

Market Integrity

The integrity of The Nasdaq Stock Market depends on free and open price competition between market makers. Some market makers can be dissuaded from competitive pricing by others through a variety of harassment methods. These methods have been used to "enforce" improper pricing conventions that can result in unfair profits to the market makers at the expense of the customer. Additionally, market makers can coordinate their pricing and trading activity to hide information from other participants and customers, who have a right to it, as a means of influencing prices. The Market Regulation Department is responsible for surveillance of the market with respect to these and any other schemes involving unfair coordination or anticompetitive behavior. Examples for which the system provides surveillance include the following: First, after receiving a large customer order to purchase a particular security, a market maker buys the stock from a second market maker, then resells it to the customer at a higher price; the customer could have purchased the stock from the second market maker at the lower price. A market maker stops receiving orders after he/she narrows his/her quotes in a security; the orders return when he/she returns his/her quotes to a more typical level.

Best Execution

The best-execution rule states that the price received by an investor should be as favorable as possible under prevailing market conditions. Such a favorable price is usually questionable when executing a trade outside the inside quotes. The regulatory enforcement of the best-execution rule is greatly complicated by market conditions, such as relative volatility and liquidity, the size and type of transaction, available communications, accessibility to the primary markets, and quotation sources that might grant exceptions to the rule.

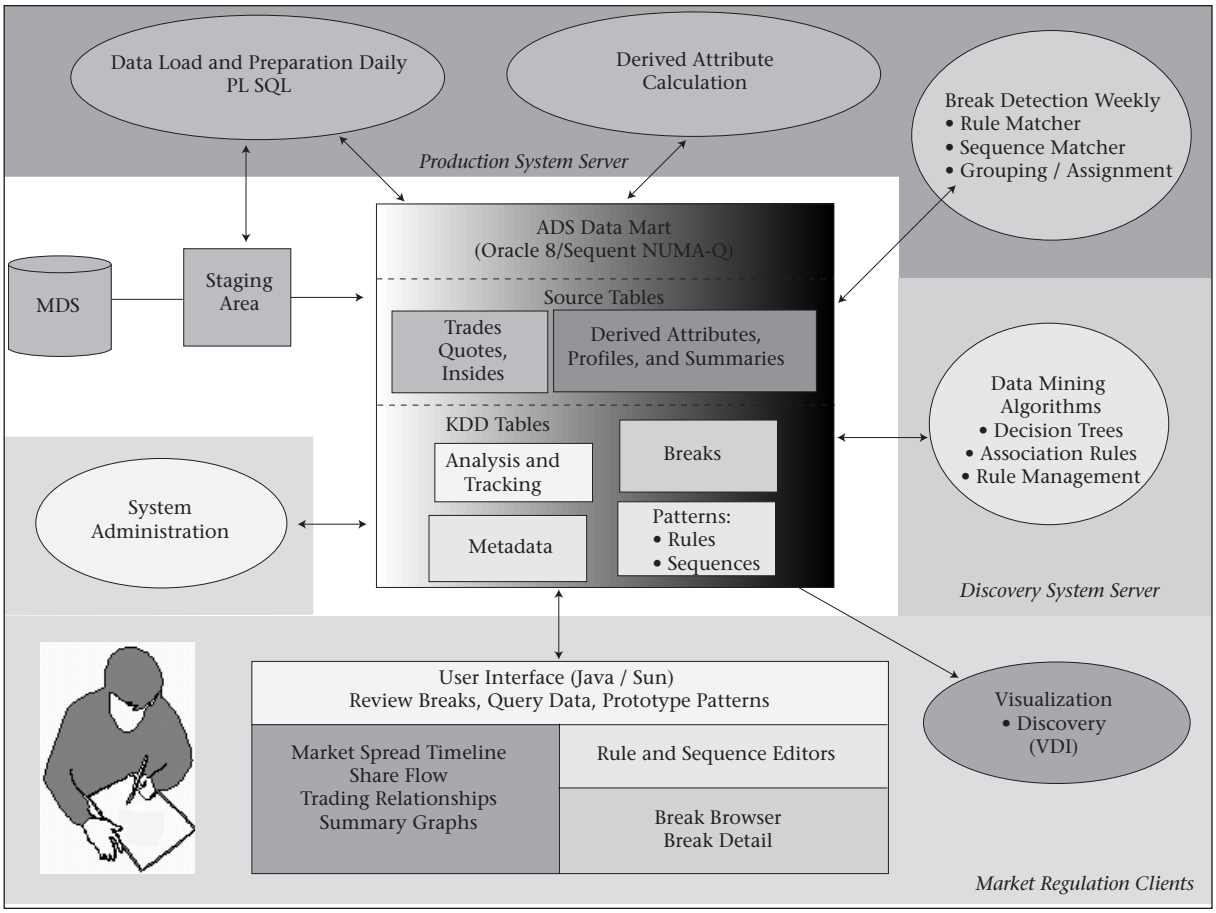

Figure 1. ADS Architecture.

Application Description

This section describes ADS, how it works and what it is. The ADS architecture is illustrated in figure 1. The key functional modules of ADS are the ADS data warehouse, detection programs, discovery programs, and the user interface.

Data Warehouse

The ADS data warehouse is the heart of the system. As of April 1998, it was about 240 gigabytes (GB). It consists of two sets of tables, referred to as *source tables* and *metadata tables*. The source tables contain market data about trades, quotes, and insides. These tables consist of attributes from systems from which ADS receives information and additional attributes and indexes that are calculated for use by other components of ADS as well as summary and profile information about issues and firms. Metadata tables hold setup and execution control parameters for the break-detection and break-discovery jobs, the rule and sequence patterns that are matched by the detection jobs, the results of the break-detection and break-discovery jobs (breaks and rules), and data necessary for various user interface components. These data provide knowledge management for patterns and an audit trail for all processes involving ADS: discovery, detection, and analysis.

Data-load and data-preparation programs update the data warehouse daily. They combine information from the source tables and calculate additional attributes deemed useful for detection and discovery. The data sources for ADS present a view of the market as a series of transactions with separate trades and price updates. Matching these transactions to produce a more coherent market state is a major computational effort but is essential to provide the derived attributes that capture market context.

Detection Programs

ADS detection programs consist of two pattern matchers—(1) a rule matcher and (2) a time-sequence matcher—that are run weekly to generate breaks. Potential violations are represented as rule or sequence templates, or *patterns,* which, when the conditions are matched by market data, result in the creation of a break.

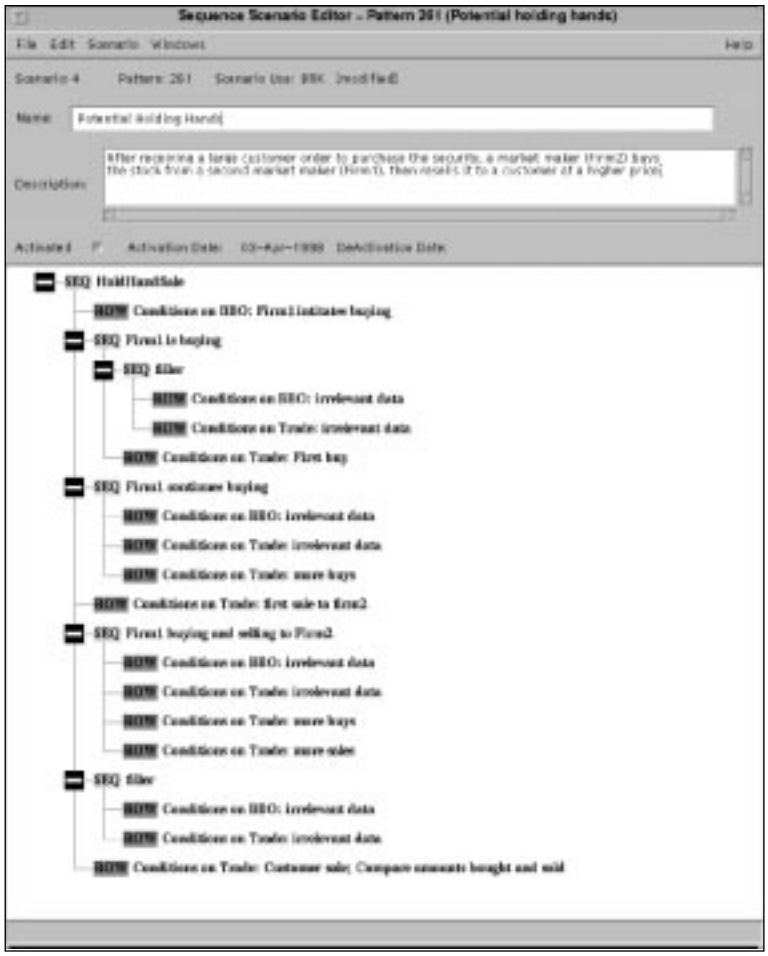

Figure 2. Temporal Sequence Pattern Editor.

Breaks are assigned to analysts through distinct automated break-assignment modules that reflect each user team's work process.

Sequence Matcher The *sequence matcher* is a program for finding instances of temporal patterns in databases. It seeks triggering events that, in turn, seek other events in either forward or reverse time sequence and order that could indicate deliberate behavior of concern. The algorithm was independently developed at SRA, drawing on work in the discovery of temporal associations (Mannila 1995).

A graphic editor (figure 2) has been developed for the temporal sequence patterns used in ADS. A typical sequence for matching a violative scenario called Holding Hands is shown (in redacted form). In this sequence, one market maker with a customer order uses a second market maker to acquire shares, hiding the true nature of these transactions from both the market and his/her customer.

Each SEQ line represents a possible loop, with user-specified time duration and repetition conditions. Each ROW line represents an SQL-style conditional expression that is matched against a row of input data. The entire pattern must match the input data for it to be accepted and a break to occur. Thus, a number of ROW elements are required to "fill" or match data that are irrelevant to the particular scenario.

In addition to matching input data, ROW clauses can bind user-specified variables to values encountered in the input data. These bindings can be preserved and displayed with the break for the analysts to consider or can be used internally to control the pattern match. Finally, the patterns can initiate many times, overlapping one another in the data. Each instance that survives by forming a complete match can result in a break. Thus, the sequence matcher can have many patterns in a partial state at one time. For example, this pattern would begin with each market maker who initiates buying activity.

The sequence-matcher algorithm works by querying the metadata tables to load one or more patterns into memory. The sequence-matcher algorithm is similar to a regular expression matcher. It maintains a list of potential match states. At each step, a row is fetched, and a new state is started for each pattern. Existing states are advanced if they match the constraints on the pattern location where they are currently. When a state reaches the end of a pattern, it is a match. The sequence matcher can be run in forward or reverse mode, fetching rows in increasing or decreasing time order depending on whether the triggering event for the sequence occurs before or after the other necessary conditions. In a single pass, multiple tables can be scanned for several patterns concurrently. The sequence pattern language uses a syntax and precedence similar to the C programming language.

There are three types of input to the sequence matcher: (1) target configurations, (2) patterns, and (3) call-back functions. A *target configuration* details the database columns to be processed along with any row-ordering conditions. *Patterns* are specified either from the metadata tables or a text file. *Call-back functions* are C++ functions compiled into a dynamically loadable object. They can be time filters or actions. A time filter is a way of coordinating the ordering of rows from heterogeneous tables. Output of the sequence matcher is determined by the action functions that are called when a pattern is matched. The break-generation action function populates the metadata tables with breaks. The break-generation call back inserts a new row into a table for every match found. Each matching state

has some number of rows that are the instantiation of the pattern.

The temporal sequence matcher has been the essential element for break detection in ADS. In the market integrity domain, it is used to detect entire regulatory scenarios. In best execution, it is used to detect individual violations, which are then aggregated to form regulatory breaks. The late-trade domain uses the sequence matcher to detect potential regulatory violations and identify and label late trades that correspond to specific market conditions. It has also been used to experiment with new ideas, gathering instances of patterns that might not be complete breaks but that the analysts might examine to gain insight into new market behaviors and possible new violations.

Rule Matcher The *rule matcher* fetches trade data and produces breaks based on the detection of repeated instances of predefined behaviors, represented formally as rules with an antecedent and an optional consequent. Each rule has two measures of strength, called *confidence* (fraction of rows satisfying the antecedent where the consequent is also true) and *support* (fraction of rows in the entire table that this rule holds on), which are used as parameters to ensure that breaks that are generated by the rule correspond to significant patterns of activity. An example pattern would be one that looked for firms showing a high percentage of trades involving more than 10,000 shares that are designated late.

The rule matcher internally represents each attribute that is mentioned in at least one pattern and uses the attribute names to build the query that will retrieve the targeted trade data. The patterns are represented as conjunction tests on attributes. Trees are created that contain the pattern conjunctions and counts corresponding to the number of times that the conjunction is detected.

Each record retrieved is given an internal representation that facilitates tree traversal based on attribute tests passed. If an attribute has a bind test defined for it, a new test is added whenever a new value for the attribute is encountered in the data. The record representation is given to the tree structures, and all necessary conjunction counts are updated. The output of rule matching results in storing breaks and break-related data in the database.

Discovery Programs

ADS includes parallel and scalable decision tree and association rule implementations that can be used to discover new rules reflecting changing behaviors in the marketplace.

Association Rules The *association rule algorithm* is a procedure for generating rules from a table (Agrawal et. al. 1993). Our implementation of the association rule algorithm is written in C++ and performs direct access to an ORACLE8 relational database management system (RDBMS) through a database class library invoking the ORACLE call interface (OCI) API (application program interface). Parallelism is achieved using an implementation of the message-passing interface. The algorithm uses parallelism in several places to divide up tasks. Each process is central processing unit intensive and allocates its own memory.

Attribute filters are used to reduce the number of association rules reported by the algorithm. They provide guidelines about how certain attributes make a rule interesting or not interesting. There are four types of attribute filter that add the capability to include or exclude attributes, group attributes in rules, and specify functional dependencies. These filters are instrumental in reducing the number of redundant and definitional rules generated.

Decision Trees The decision tree algorithm is a data-mining tool designed to find patterns with respect to a single specified data column called the *dependent attribute* (Quinlan 1993). The input data consist of a number of "examples," each of which is a vector of attribute-value pairs. The algorithm outputs a set of rules that use the independent attributes to predict or characterize values of the dependent attribute. These rules have a conjunction of independent attributes on the left-hand side and contain only the dependent attribute on the right-hand side.

Rules are extracted from a decision tree by tracing the path from a leaf to the root of the tree. The dependent-attribute value assigned to this node becomes the right-hand side of the rule, and the independent attributes and values in the nodes on the path to the root become the conditions on the left-hand side of the rule.

Rules are pruned by dropping conditions from the left-hand side and seeing if a better rule is produced. If a rule has N conditions in the left-hand side, then dropping each condition can produce N possible alternative rules. Each of these rules is evaluated, and if any are better than the original rule, then the best alternative is chosen, and the process is repeated. A rule's quality is determined using its normalized estimated net worth. Once each rule is pruned, duplicates are removed.

Rule Management A run of decision trees or association rules against a data sample of one firm or one security usually produces several rules; thousands of rules can result from

ADS includes parallel and scalable decision tree and association rule implementations that can be used to discover new rules reflecting changing behaviors in the marketplace.

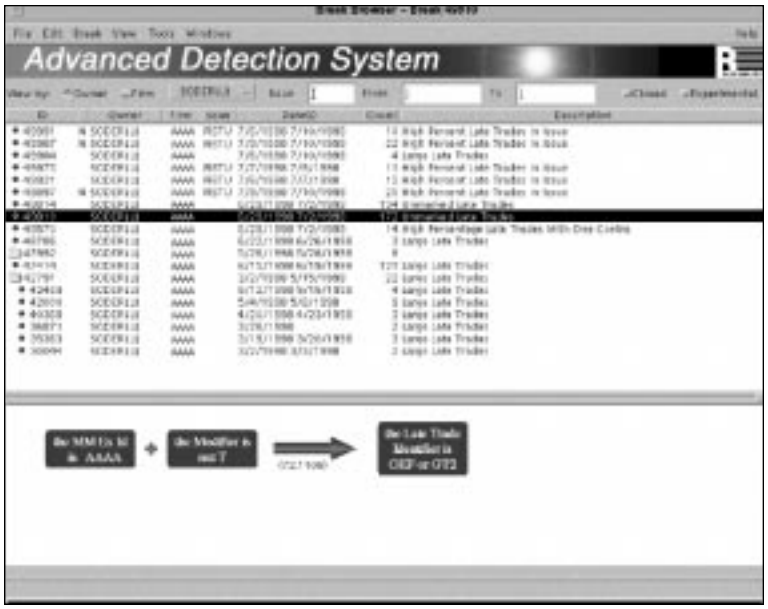

Figure 3. Break Browser (Late-Trade Domain).

runs against all securities and firms, necessitating automated rule-management capabilities. Moreover, to follow dynamic behavior of the market, the pattern base must be highly adaptive, allowing additions, refinements, and deletions. The *rule-management module* supports transformations of discovered rules to break detection patterns, similar to the approach described in Fawcett and Provost (1997). Rule management comprises three operations: (1) filtering, (2) refinement, and (3) deletion.

During filtering, discovered rules are pruned to a subset that is unique (not repeated among themselves), is new (different from existing patterns and rules), has high confidence, and has good support in data. Filtering compares the generality of two rules by examining the similarity of the rules' conditions and conclusion (Tecuci and Duff 1994; Michalski 1983). The comparison algorithm uses domain-specific and domain-independent heuristics to capture the meaning of various rule attributes. The filtering process decreases the number of rules from thousands to hundreds, which are still firm or security specific.

During refinement, rules obtained from the filtering process are refined by generalizing their conditions and ranking them against random data samples. The refinement process includes generalization of rules by dropping market participant- or security-specific information according to the specified refinement heuristics (Dybala 1996) and ranking and testing the obtained rules against various data sets to refine their support and confidence thresholds. Finally, the best-performing rules are selected. Rules that pass the refinement process are unique, general, and different from existing rule patterns. The result of the refinement process is several or possibly tens of new rules that are presented to the analysts for a review and possible promotion to active break-detection patterns.

User Interface

The ADS user interface consists of screens for break processing and management, tabular displays for viewing detailed trade and quotation information associated with a particular break, two-dimensional graphic displays that allow an analyst to easily visualize market activity at the time of a break, and three-dimensional displays that are useful for viewing large aggregates of information.

The *break browser* (figure 3) is the user's main window for reviewing and analyzing breaks generated by ADS. In the top half of the window, a number of standard data elements are displayed and can be used for filtering and sorting the analyst's set of open breaks. Note that some breaks are identified by folders. These are composite breaks, formed by either manual or automatic grouping of machine-detected breaks. For example, in the best-execution domain, breaks are automatically grouped by firm and are presented to the analyst as a set of related behaviors by one trader. Breaks can also be assigned automatically or manually to individual analysts, depending on the business process of the market-regulation team involved.

The lower section of the break browser is used to display a detailed description of the selected break. Breaks are of two types, depending on the method used to detect them. In figure 3, we see a rule break, displayed as a set of conditions with the number of data rows matching the full rule and the left-hand side only. Sequence breaks are more complex, so they are displayed as descriptive text together with a series of variables and bound values, specific to the particular pattern that resulted in a selected break.

The rule analyzer (figure 4) is an ad hoc rule match function that allows the analyst to try out variations of existing rules over limited portions of the database in an interactive manner. Figure 5 shows the details of market trades, with right-hand-side matches highlighted, which were matched by this rule. The rule analyzer was constructed for experimentation with potential rule patterns; however, it has also been used as an ad hoc query tool and has resulted in the identification of specific

cases of regulatory interest.

These screens, as well as a number of other detail displays, such as daily aggregates of trading and quoting behavior, are available from the break browser in tabular displays. They allow the analyst to investigate and later substantiate the particulars of a break when developing a case for possible regulatory action.

The market context around the time of a suspected violation in a security is determined by three things: (1) the trades in the security, (2) the bids and asks of the market makers in the security, and (3) the inside bid and ask. The difference between a bid and an ask price is called the *spread*. This context is captured visually in the market-spread timeline (figure 6). A market maker's bid and ask in an issue are plotted against time in one color and the inside bid prices and ask prices in another. Trades are displayed as dots. Details of trades and quotes are available through a drill-down capability. The visual display allows analysts to quickly identify important events, such as the inside spread being narrowed, multiple small trades being executed against a market maker, or a market maker buying up a lot of shares in an issue.

The share flow display (figure 7) shows a group of trades in order by execution time. Each trade is represented as an arrow from the buyer to the seller marked with the number of shares and price. When a group of trades is displayed by time, patterns of share flow are visible. A firm can collect shares through multiple medium-sized buys and sell them in one large sale. Shares can pass through multiple firms before reaching their final destination. These patterns of share flow are crucial for analyzing potential violations.

Trading relationships between broker-dealers can be a key to understanding the context surrounding a break. The analyst can display these trading relationships as linkages (figure 8) aggregated over the set of trades involved in a break. This display and the share flow display mentioned previously are also available from within the market-spread timeline mentioned earlier. In this case, the analyst can select any set of trades that might be of interest and pass them to either display.

Detecting any potential conventions or regularities in market behavior that might indicate improper coordination by market participants requires the ability to rapidly review large amounts of market data for patterns and anomalies. To provide this ability, Visible Decisions Inc. DISCOVERY software was selected for three-dimensional displays (Martin 1996). Two three-dimensional landscapes have been developed. The pricing landscape (figure 9) displays a large amount of summary data regarding quotation activity by multiple market makers in many issues. It highlights the possible pricing conventions in portions of the market. These conventions can be agreements among market makers to quote only in specific price intervals. If these agreements are enforced by peer pressure or harassment, they become anticompetitive and are violative practices. A visual examination of this display permits an analyst to rapidly determine patterns of quotations in an issue and similarities between the quotation behavior of different market makers. Situations where the pricing conventions hold for all but a few market makers are those most likely to result in anticompetitive behavior and warrant the closest review. In addition, new conventions are more

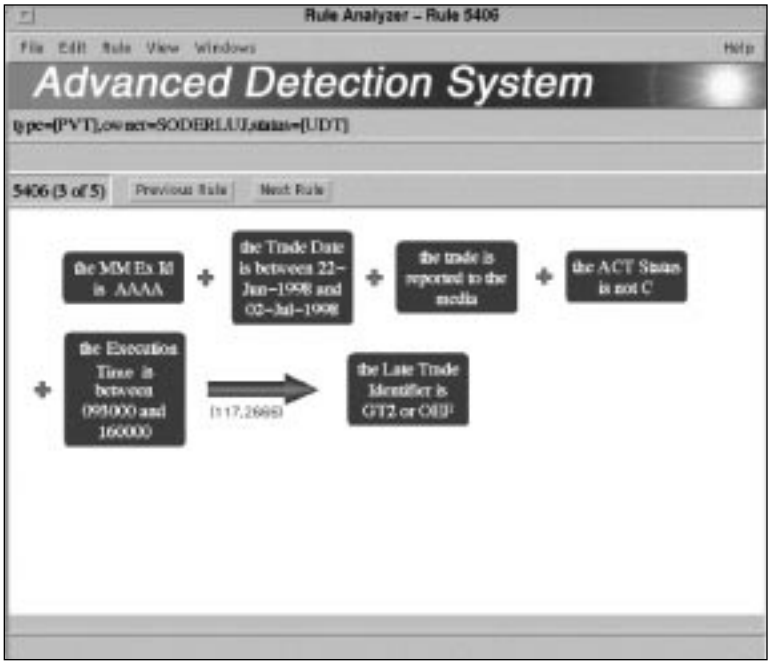

Figure 4. Rule-Pattern Analyzer.

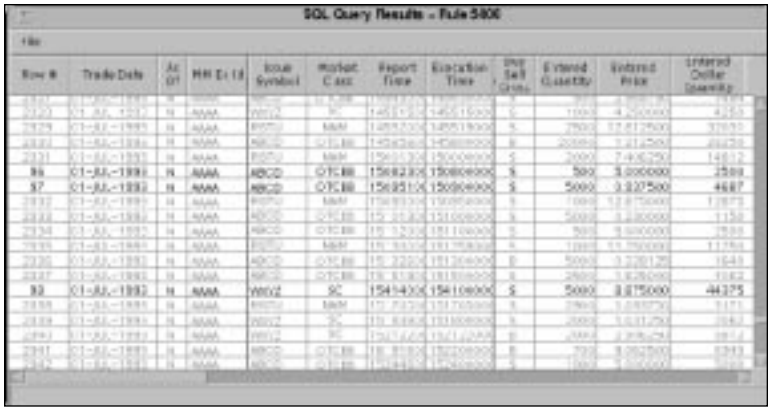

Figure 5. Trade Details Tabular Display.

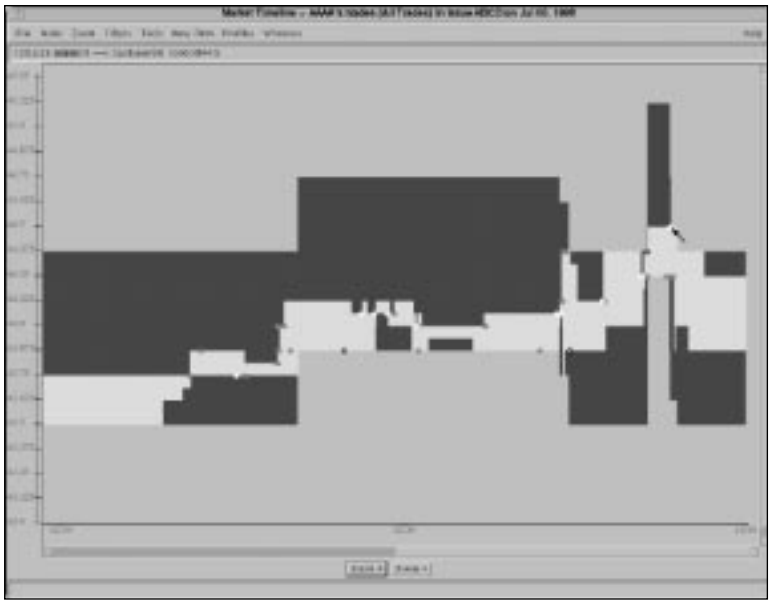

Figure 6. Market-Spread Timeline.

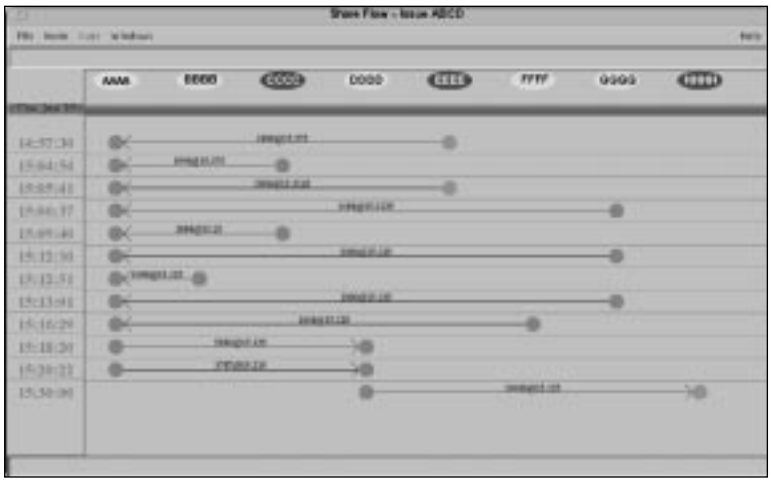

Figure 7. Share Flow Display.

easily visible because the display can be adjusted by various thresholds and filters.

The second display, called a *spread landscape* (figure 10), allows an analyst to view the quotation and trading behavior of a set of market makers in a particular issue over a specified time period. This display can be viewed as a generalization of the two-dimensional spread display described previously.

In addition to these graphic and tabular displays, a number of specialized reports have been included in the production system to enable rapid analysis and documentation of breaks. These productivity-enhancing features are essential in a working system used regularly by multiple teams of analysts.

Hardware and System Software Environment

ADS system hardware currently consists of a 12-processor Sequent NUMA-Q host computer acting as the production database server. Scalability and growth were significant factors in the choice of a server platform. Sun SPARCSTATIONS serve as user workstations running the ADS client as well as other applications. A second, smaller Sequent NUMA-Q is used as a development and knowledge discovery platform, and a variety of file servers are installed on Sun SPARCSTATIONS. The operating systems are varieties of UNIX. ORACLE 8.0.4 is the RDBMS. The ADS client is written in JAVA, which has proven to be an effective language for both prototyping and final implementation.

Uses of AI Technology

ADS integrates data-mining (decision trees and association rules), pattern-matching (rules and time sequences) and visualization techniques in a single large-scale application, as detailed in previous sections. It represents an advance over previously reported applications because of the large scale of the application, the combination of discovery and detection components with the ability to discover new rules and promote them into the detection system, the explicit representation of rules in the database for use in detection and as a result of discovery, the development of the time-sequence pattern matcher, and the direct database-access parallel implementations of the detection and discovery components.

Commercial Tool Evaluation

We evaluated off-the-shelf knowledge discovery and data-mining tools during the proof-of-concept phase of ADS development. We surveyed more than 20 off-the-shelf knowledge discovery database products and conducted a detailed evaluation of two of them. However, none of them appeared to meet our needs for a system that would do the following, so we developed the components as part of a custom application:[1] (1) function in NASD's hardware and software environment; (2) have an open architecture for integration with other necessary system components, especially the database management system and the (then) to-be-developed work-flow–management components; (3) function in a highly dynamic environment; (4) integrate a comprehensive variety of methods; (5) provide intermediate results to all components; (6) scale to handle production volumes of over one million trades and quotes a day; and (7) most importantly, provide an integrated struc-

ture for ongoing detection of improper activity combined with analysis to identify new patterns of potential regulatory interest.

Related Applications

The FINCEN AI system (FAIS), described in Senator et. al. (1995), motivated some of the basic ideas of ADS, although the specific requirements differ. Both are instances of a type of fraud-detection system called break-detection systems that are described in Goldberg and Senator (1997), which also contrasts the two domains along several characteristics. ADS advances over FAIS along several dimensions, in particular, its much larger data volume, its incorporation of automated discovery techniques for identification of new patterns of potential regulatory interest, its explicit representation of multiple domains and user groups, and its use of a time-sequence pattern matcher for detection.

Fawcett and Provost (1997) describe an approach to cellular telephone fraud detection that automatically learns indicators of potential fraud from a large database of transactions. It uses the indicators to create monitors that profile legitimate behavior and detect anomalies as outliers. The output of the monitors are combined to generate alarms; just as in ADS, repeated occurrences of alarms are aggregated to identify fraudulent patterns or practices.

Application Use and Payoff

The development of ADS has taken members of the Market Regulation Department to a new level in monitoring the markets it is charged to monitor and regulate. For a number of years, they have had systems that point to potential instances of regulatory concern on market data. In addition, they have a history of taking disciplinary or enforcement action on those on which action was warranted. However, prior to ADS, we did not have a system to help us detect potential violative patterns and practices in quotation and trade data. We depended on human analysts to recognize patterns and practices as a series of activities became known to them. Now, we search proactively through all our quote and trade data for these patterns or practices of regulatory concern. Individual instances of violative activity can be explained away by a perpetrator, but strong evidence of a pattern or practice is more difficult for a perpetrator to defend.

What is the payback? First, we have seen notable success in the area of late trades, the first of the areas of concern targeted by ADS. We have increased the hit ratio of good breaks, or leads, to overall breaks by a factor of three over the best of our prior approaches. Thus, the analysts spend less time on efforts that expose no regulatory concern. ADS permits all the analysts, those with much experience and skill and those with less

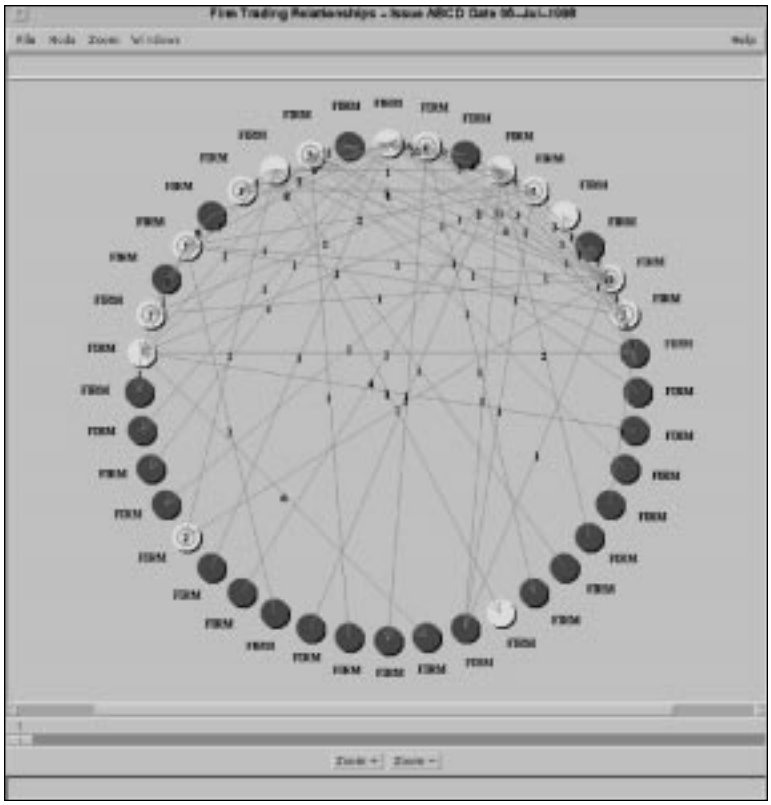

Figure 8. Trading Relationship Display.

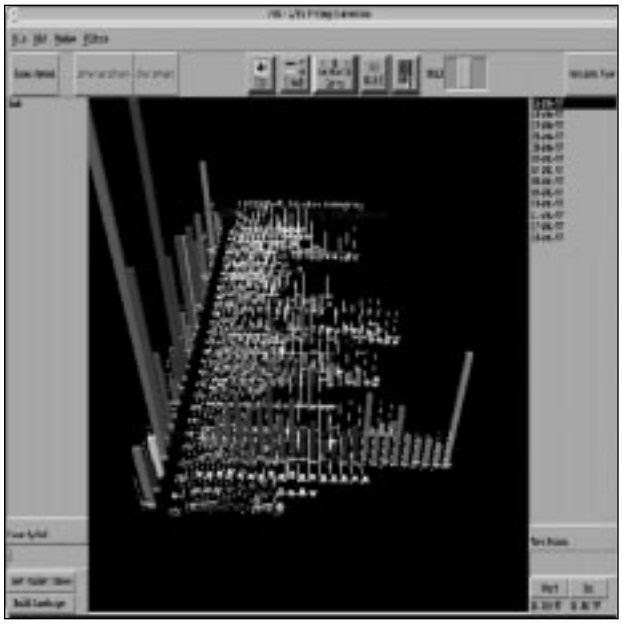

Figure 9. Pricing Landscape.

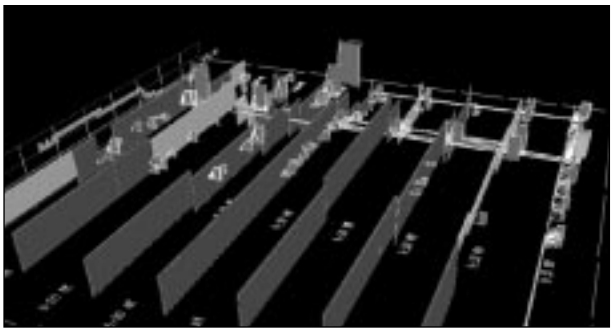

Figure 10. Spread Landscape.

experience and skill, to be effective at a higher level. It enables new analysts to develop a level of sophistication much more quickly—about half the time. We have been able to bring many more actions in this area than previously. Second, we have been able to establish a new team to monitor concerns in market integrity. We have been able to point to potential anticompetitive, harassment, or collaborative activity in a way not previously possible. This team is now able to proactively surveil the market instead of react to customer complaints. Third, we have started a pilot in a third area to monitor firm responsibilities for best execution. We have had this kind of monitoring in the past but not to detect patterns and practices. Fourth, we have the visualization tools that permit the analysts to "see" things that our processing might miss. Fifth, we have a system that can be adapted quickly to new market realities or changes in activity. Sixth, we monitor the data thoroughly. We do not sample. We run tests exhaustively over all data, which has required quick algorithms and quick machines to handle a truly large amount of data. The use of ADS has also resulted in the discovery of additional knowledge about market behavior in each of the three domains implemented to date.

From June 1997 through August 1998, ADS generated over 10,000 breaks in the market-integrity and late-trade domains that have resulted in more than 1,000 follow-up actions of various types (such as requesting records from the securities traders involved or referring to other units of NASD Regulation). This rate, over 10 percent, is a factor of 3 increase over previous break-detection systems. Another 80 percent of the breaks have been closed, yielding valuable experience that has been reapplied to the detection process. The new domain, which addresses violations of the best-execution rule, is producing about 3000 breaks a month. These breaks are aggregated by trading firm on a quarterly basis, resulting in about 600 firm reports a quarter. Between 5 and 10 percent of the most violative firms are contacted for further investigation. The apparently high "false-alarm" rate is acceptable because breaks can be closed quickly when no action is warranted and because the trade-off cost of missing a real violation is extremely high, corresponding to a zero tolerance for major violations. Further, the process of investigating a break, even a false alarm, is a significant deterrent to violative behavior and typically results in market improvements.

A key payback for a surveillance system is the degree of coverage in terms of the number, type, and detail of potential violations we can monitor and the amount of market data we can review. Our initial estimates of improved surveillance coverage with ADS compared to manual surveillance is a factor of about 225. We have also seen 75-percent reductions in the complexity of some surveillance protocols (corresponding to 300-percent productivity improvements) as well as significant reductions in potential violations in the areas of both late-trade reporting and market integrity, corresponding to an improved market for all investors.

Another measure of the coverage by ADS is the amount of knowledge in the system and its continual refinement. Figure 11 shows the knowledge embodied in the rule and temporal sequence patterns expressed as the number of clauses (either rules or data-row conditions) activated and deactivated each month that the system has been in production. Continual refinement can result in either the addition of more specific conditions or the removal of overly restrictive conditions, but the overall trend is toward more and more effective knowledge.

Additional up-to-date information regarding specific regulatory actions taken by NASD Regulation, including those resulting from ADS, can be found at www.nasdr.com, which describes disciplinary actions in all areas of NASD Regulation's jurisdiction.

Application Development and Deployment

The ADS project team consisted of staff from NASD Regulation, Inc. (Office of Technology Services and Market Regulation Department); NASD Production Services Department; and SRA International, Inc. At its peak, the team consisted of approximately 22 people. Table 1 lists key ADS development milestones.

ADS began as a proof of concept in the area of late-trade reporting. The project was initiated and a team assembled in April 1996. Scenarios

and corresponding patterns were identified, and rule-matching and rule-discovery algorithms were developed. A demonstration that this approach could be effective in improving late-trade surveillance resulted in a July 1996 project steering committee decision to proceed with a pilot on live data. This pilot was deployed in October 1996. During the summer of 1996, the market-integrity area was determined to be of key importance in resolving concerns addressed in the SEC 21 (a) report, and the decision was made to expand the late-trade pilot into ADS during 1997. The time-sequence matcher was developed during 1997 for the needs of market integrity and for some late-trade scenarios that could not adequately be represented by rules. ADS went into full production on 31 July 1997.

ADS was one of the first production implementations of ORACLE 8, undergoing conversion within a month of becoming operational. Because of the success of late-trade reporting and market integrity, it was decided to add the best-execution domain. Work began in late 1997 to develop patterns and scenarios. Continued improvements in functions based on feedback from the users and in the meeting of NASD Production Services standards for a well-documented and predictable system was a major emphasis in late 1997 and early 1998. Release 2.0 in May 1998 formally recognized the distinction between the ADS application and SRA's knowledge discovery database EXPLORER TOOLKIT, which serves as the underlying discovery and detection engines. Quarterly releases continue to add additional business functions as required.

The separate domains came online at different times. Each domain began as an operational pilot, with live data-generating experimental breaks, so the patterns could be tuned appropriately and analyst feedback could be incorporated into the development. The late-trade domain analysts have been evaluating breaks since October 1996 as a pilot project. The market-integrity team began work in July 1997, with the entire system beginning full production accountability in August 1997. An application of ADS to the enforcement of the best-execution rule began as a pilot in January 1998 and was upgraded to production in May 1998. Two additional domains were included in late 1998 as part of release 98.3. They address regulatory violations involving electronic communications networks (ECNs) and automatic tracking of quotes on other exchanges.

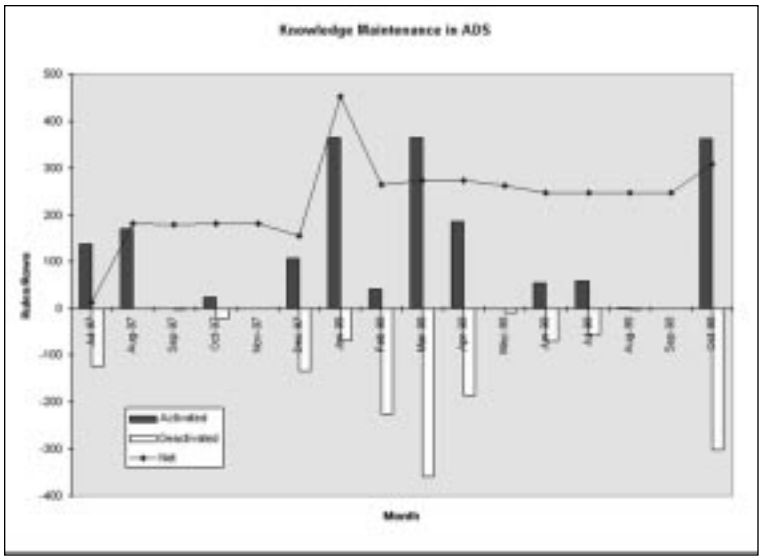

Figure 11. Pattern Knowledge in ADS.

Real-time monitoring of The Nasdaq Stock Market is one of the ways the National Association of Securities Dealers, Inc., protects the investor.

Knowledge Maintenance

Because ADS applies to multiple dynamic domains, knowledge maintenance is a key issue. Knowledge maintenance is enabled by processes and tools. Weekly meetings are held with key users in each domain area to review current breaks and pattern performance. At these meetings, new scenarios are discussed, and prototype patterns are evaluated for inclusion in the system. Tuning of operational parameters is done at these reviews so that the quantity of breaks is consistent with the analysts' ability to evaluate them. As break quality improves, thresholds can be adjusted to allow more marginal breaks as well as allow new patterns to be detected.

We release new break-detection patterns into the production job stream on a monthly basis to provide the opportunity for flexible responses to new market conditions or regulatory priorities while we maintain the ability to plan our production operations. Modifications to existing patterns that do not affect production operations can occur weekly. Patterns that require new data elements or new displays are released in conjunction with the quarterly software releases. Patterns can be released to the production environment in an experimental status to allow for evaluation and final approval prior to being included as part of the regulatory process.

A combination of manual and automated discovery was anticipated, with the emphasis early in the project on manual specification of

April 1996	Late-trade reporting: Project initiation
July 1996	Late-trade reporting: Proof of concept
August 1996	SEC 21(a) report
October 1996	Late-trade reporting: Pilot (release 1.0)
January 1997	Market integrity: Project initiation
June 1997	Late-trade reporting— Production and market integrity: Pilot (release 1.1)
August 1997	Market integrity: Production (release 1.2)
September 1997	ORACLE 8 conversion
September 1997	Best execution: Project initiation
January 1998	Best execution: Pilot (release 1.3)
May 1998	Best execution: production (release 98.1)
July 1998	Release 98.2
December 1998	Release 98.3

Table 1. ADS Development Milestones.

detection patterns to jump start the project and make use of the Market Regulation Department's expertise. As ADS matures and we reach the limits of what analysts already know, we expect automated discovery to play an increasing role in shaping the system's knowledge component. We have created a separate instance of ADS for pattern-development purposes. This instance is used to prototype and evaluate new pattern specifications that might result from domain knowledge or automated discovery. We also use this environment to evaluate the performance of the break-detection jobs that use new patterns. This separate environment is necessary because the daily operations of production jobs must be predictable to allow all applications to function in our environment. During the pattern-development process, we found several situations of potential regulatory interest that resulted in actions even before the particular pattern was included in the production application.

The ADS application includes tools to introduce new and modify existing rule and sequence patterns. The sequence editor provides a graphic structure for modifying the more complex specifications of multi-input, nondeterministic temporal patterns.

The ADS user interface provides features for managing experimental patterns, which are run in production but are not yet validated as producing breaks of regulatory value. The use of experimental patterns allows us to evaluate newly proposed or modified patterns on current data, ensuring they stay current with market conditions. Finally, we anticipate the regular addition of new domains to ADS, requiring new sets of patterns corresponding to the types of potentially violative behavior of interest.

Acknowledgments

We thank all our colleagues at NASD and SRA who aided in the development of ADS or contributed to its knowledge bases, especially our executive sponsors and our Steering Committee. Most important, we thank Jim Cangiano, senior vice-president of Market Regulation, the project sponsor, whose vision, leadership, and support made the project not only possible but also successful. Other Steering Committee members included Vice-Presidents Bill Bone, Seth Chamberlain, Tom Gira, Sam Laughery, Derek Linden, Steve Luparello, David T. Miller, and Ed Morgan and Senior Vice-President Emerson Thompson of SRA International, Inc. Market Regulation staff members who contributed include Jim Bohlin, Fernando Cabrejo, Patti Casimates, Mario Dieudonne, Jim Dolan, Tonia Edwards, Patrick Geraghty, Trudy Hanbury, Ruth Kapusta, Tom Kober, Eric Lamb, Jim Lamke, Holly Lough, Amrita Mattoo, Joe McDonald, Kevin McEvoy, Chris Phillips, Carol Russo, Steve Simmes, Jon Soderlund, Barry Postell, Blair Vietmeyer, Peter Virador, Shelly Wilson, and Bob Yabroff. NASD Technology staff contributors include Patricia Casillas, Rosemary Casteel, Lowell Cooper, Gina Davis, Bob Dasher, Jim Gallalee, Len Gatrell, Charles Jackson, Cindy Kienzler, Randeen Klarin, Roja Kolachina, Dennis Lee, Steve Nieberding, Dennis Phillips, Yelena Pruzhanskaya, Vladimir Sazonov, Henry Smith, Ge Wang, and Aung Aung Win. SRA International staff contributors include Scott Bennett, Bill Brooks, Mariann Bryant, Brad Christiansen, Steve Donoho, Matt Fluet, Linda Hagen, Peter Halverson, Sue Jones, Craig Lovell, Bruce Megahan, Deepak Misra, and Bruce Redmon.

Note

The authors of this article are employees of the National Association of Securities Dealers (NASD) Regulation, Inc., or its contractors. The views expressed here are those of the authors and do not represent an official policy statement of NASD Regulation, Inc.

References

Agrawal, R.; Imielinski, T.; and Swami, A. 1993. Mining Association Rules between Sets of Items in Large Databases. In Proceedings of the ACM SIGMOD Conference on Management of Data, 207–216. New York: Association of Computing Machinery.

Dybala, T. 1996. Shared Expertise Model for Building Interactive Learning Agents. Ph.D. thesis, School of Information Technology and Engineering, George Mason University.

Fawcett, T., and Provost, F. 1997. Adaptive Fraud

Detection. *Data Mining and Knowledge Discovery* 1(3): 291–316.

Goldberg, H. G., and Senator, T. E. 1997. Break-Detection Systems. In *AI Approaches to Fraud Detection and Risk Management: Collected Papers from the 1997 Workshop,* 22–28. Technical Report WS-97-07. Menlo Park, Calif.: AAAI Press.

Mannila, H.; Toivonen, H.; and Verkamo, A. I. 1995. Discovering Frequent Episodes in Sequences. Paper presented at the First International Conference on Knowledge Discovery and Data Mining, 20–21 August, Montreal, Canada.

Martin, J. 1996. Beyond Pie Charts and Spreadsheets. *Computerworld* 30(22): 37.

Michalski, R. S. 1983. A Theory and Methodology of Inductive Learning. In *Machine Learning: An Artificial Intelligence Approach, Volume 1,* eds. R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, 83–134. Palo Alto, Calif.: Tioga.

Quinlan, J. R. 1993. *C4.5 Programs for Machine Learning*. San Francisco, Calif.: Morgan Kaufmann.

Senator, T. E.; Goldberg, H. G.; Wooton, J.; Cottini, M. A.; Umar Khan, A. F.; Klinger, C. D.; Llamas, W. M.; Marrone, M. P.; and Wong, R. W. H. 1995. The Financial Crimes Enforcement Network AI System (FAIS): Identifying Potential Money Laundering from Reports of Large Cash Transactions. *AI Magazine* 16(4): 21–39.

Tecuci, G., and Duff, D. 1994. A Framework for Knowledge Refinement through Multistrategy Learning and Knowledge Acquisition. *Knowledge Acquisition Journal* 6(2): 137–162.

J. Dale Kirkland is a project manager in the Market Regulation Department of NASD Regulation. He has been an employee of NASD since 1988 and a member of the Market Regulation Department since 1990. He initiated the business and technology development of ADS in late 1995 and was the original project leader and subsequently business program manager for ADS. He has contributed to the development and support of other market surveillance break-detection systems. Prior to his employment at NASD, he performed contract work for the U.S. Department of Defense. He is also a former math teacher and department head. He has degrees in mathematics, computer science, and school administration and supervision. His e-mail address is kirkland@nasd.com.

Ted E. Senator is the director of the Knowledge Discovery in Databases (KDD) Group in the Office of Technology Services of NASD Regulation. He holds degrees in physics and electrical engineering from the Massachusetts Institute of Technology. He is a candidate for the Master of Science in finance degree from the George Washington University and has done additional graduate work in physics and computer science. He is a three-time winner of the American Association for Artificial Intelligence Innovative Applications Award for ADS and for previous work for the U.S. Department of the Treasury and the U.S. Department of the Navy. He has served on the KDD program committee and the Innovative Applications of Artificial Intelligence program committee, which he chaired in 1997. His technical interests include AI applications, KDD, and intelligent systems engineering, especially as applied to finance and fraud detection. His e-mail address is senatort@nasd.com.

James Hayden is the program manager for knowledge discovery solutions at SRA International. He has worked for more than 13 years on designing, developing, and implementing very large database–related solutions for Fortune 100 companies. He is currently responsible for defining and delivering knowledge discovery–based solutions to the commercial and federal sectors. These solutions are focused on the application of advanced pattern discovery and matching techniques to large databases to deter fraud and risk and to better understand customers. His interests are in the areas of database technologies, online analytic processing and data mining, and their impending convergence. His e-mail address is jim_hayden@sra.com.

Tomasz Dybala is a senior knowledge discovery in databases specialist at NASD Regulation, responsible for the best-execution domain of ADS. He holds a Ph.D. in information technology (1996) from George Mason University (GMU) and an M.S. in computer science (1985) from Stanislaw Staszic University, Poland. At GMU, he participated in the development and application of the DISCIPLE shell for building intelligent agents with learning capabilities. Prior to coming to the United States, he worked as researcher-instructor and a software engineer for several leading Polish research and educational institutions and electronics corporations. Tomasz has published over a dozen journal and conference papers in the areas of machine learning, applications of knowledge-based technologies to engineering design, and control engineering. His current interests are intelligent agents with learning capabilities, integrated knowledge engineering and software-engineering environments, and knowledge discovery and data-mining systems. His e-mail address is dybalat@nasd.com.

Henry Goldberg is a senior knowledge discovery in databases specialist at NASD Regulation, responsible for the market integrity domain of ADS. He is working on his second major system for detecting fraud in financial data, having come from the Financial Crimes Enforcement Network (FINCEN) of the U.S. Treasury Department where he was a principal designer of a system for detecting financial crimes. Prior to FINCEN, he worked as a senior research scientist at the Federal Judicial Center. He holds a B.S. in mathematics from the Massachusetts Institute of Technology and a Ph.D. in computer science from Carnegie Mellon University, where he worked on the HEARSAY-II Speech Understanding Project. His current research interests include AI techniques for link analysis and discovery of temporal patterns in data. He served as the cochair for the fall 1998 American Association for Artificial Intelligence Symposium on AI and Link Analysis. His e-mail address is goldberh@nasd.com.

Ping Shyr is a senior knowledge discovery in databases specialist at NASD Regulation, responsible for the late-trade–reporting domain of ADS. Hs has worked in the area of AI applications for the past 13 years. His is also a Ph.D. student in the School of Information Technology at George Mason University (GMU). His doctoral dissertation concerns multistrategy learning and knowledge acquisition and is closely associated with research projects in the Learning Agents Lab at GMU. His e-mail address is shyrp@nasd.com.