EASy: Expert Authorizations System

Jonathan Altfeld
Senior Knowledge Engineer
Brightware, Inc.
1080 Holcomb Bridge Rd. Ste 300
Roswell, GA 30076
jonathan@brightware.com

Douglas E. Landon, Ph.D.
Senior Systems Analyst
Equifax Check Services
5301 W. Idlewild Ave.
Tampa, FL 33634
landon@packet.net

Charles J. Daniels
Lead Programmer/Analyst
Equifax Check Services
5301 W. Idlewild Ave
Tampa, FL 33634
repoman@cftnet.com

Abstract

Equifax Check Services provides retail merchants and other businesses with quality decisions concerning the acceptability, risk, or fraudulence of customer checks. The greatest percentage of these decisions are provided automatically through on-line links with point-of-sale terminals. When a transaction is suspect, a referral notice is generated directing the merchant to call one of Equifax Check Services' authorization centers for additional processing. This processing considers a wide variety of information unavailable through online processing, thereby giving consumers the greatest possible benefit of doubt prior to declining checks. These high-risk authorizations had historically been handled using a legacy mainframe system involving a high degree of manual intervention. Authorizations agents would complete a lengthy, rigorous training regimen, and be monitored as to their performance. Pursuit of service excellence caused Equifax, in conjunction with Brightware Corporation, to develop the Expert Authorization System (EASy), a rule-based solution for check authorizations that uses an innovative twist on a standard blackboard architecture. EASy was deployed and is used today by as many as 300 concurrent users. By encapsulating extensive domain knowledge, EASy has effectively eliminated authorization errors, provided consistent and replicable decisions, reduced elapsed time to a decision, and reduced the average agent training time from 4-6 weeks to 3 days.

Problem Description

The original check authorization system was Tandem-based, utilizing 3270 screens to provide authorization agents with various types of information. These Agents would enter some information, then page through several 3270 screens in order to accumulate a critical mass of information about the check-writer and the transaction. As this information built up, the agents would apply decision rules to specific circumstances and deliver an authorization decision. Agents were required to follow suggestions from the system, know when to ignore or override the system, and know when specific exceptions applied. Trained agents knew which screens to view in the appropriate order, and they knew how to scroll through historical information of various kinds to identify concerns which might affect an approval decision. Even with QA monitoring and established procedures, there was still agent variability in the approval process.

In addition, the existing system was experienced as being both difficult to learn and inflexible. From four to six weeks of extensive group and individual training was required. This was followed by a period of high QA monitoring to ensure proper decision making. Intermittent long-term QA monitoring was required to maintain high quality authorization decisions.

Since training required so much time and effort, coordinating the hiring, training, and QA monitoring schedule was a significant challenge, particularly during the holiday season, when the number of agents more than doubled.

Maintenance and reliability issues were concerns with the Tandem-based system. The legacy system had expanded over many years to account for new functionality and legal restrictions, utilizing expensive and increasingly outdated computing systems. Additionally, manpower was limited for Tandem support and was becoming increasingly more expensive. Continued software and hardware maintenance was producing diminishing returns on this system. Agents were using 286 model diskless PC's, which supported 3270 emulation into a Mainframe and a Tandem, and in some cases over a Novell LAN.

Objectives of the Expert Authorizations System

Equifax Check Services needed a way of standardizing authorization decisions as well as exception-handling. The complex nature of these decisions lent themselves perfectly to Expert System technology, and posed significant problems for a procedural and/or mainframe-based approach in designing a replacement system. A decision was made that there was little or no room for error, and a great need was realized for a malleable solution with more centralized control over authorization decision-making.

There were a number of objectives for the replacement authorization system, to include:

- Standardization and automation of all types of check authorization decisions across the entire agent pool.
- Migration from reliance on legacy mainframe systems to an open and cost-effective PC/LAN-based client-server solution.
- Replacement of a limited, inflexible system with an intelligent, more easily modifiable, and flexible system.
- Reduction in call times (i.e. time to make decisions).
- Reduction of training time, complexity, and costs, combined with improvement to the training process.
- Elimination of risk due to improper decisions.
- Enabling of customer-specific processing.

Previous Similar Work

At first glance, EASy needed to accomplish much of what had been accomplished with the American Express Authorizer's Assistant (AA) expert system, but with a few notable differences.

The following are some similarities between the AMEX AA and EASy:

- Both needed to apply business policies towards on-line incoming transactions in order to reach a faster, more consistent and reliable approval decision.
- Both worked in conjunction with authorizations agents in order to provide a human interface to their customers.
- Both systems handled the anomaly decisions through agent intervention.
- Both systems rely on alternative on-line systems to handle all of the straightforward decisions; this represents in both cases a significant percentage (a proprietary, variable number). Those transactions which require more attention are referred in both cases to AA and EASy, respectively.

The following are some differences between the two systems:

- AA is a credit granting system which extends credit, while EASy is a check authorizations system. No credit is granted because the intention in EASy is to guarantee the likelihood of existing funds availability through a combination of factors. AA already has a significant portfolio of information on their customers which originates with a credit application, and builds through repeated use of the card. EASy, by contrast, can have more or less information to work with, depending on the consumer's identification and the nature of the transaction.
- AA's customers are the consumers. EASy's customers are the merchants who require check guarantees for their customers (the consumers) payment.
- While AA requires flexibility and intelligence on the part of their agents, allowing decision-override capabilities under specific circumstances, all of EASy's check authorizations decisions occur within the expert system. The EASy agent guides the customer through providing appropriate information as EASy requests it, and relays the decision to the customer.

Application Description

The replacement system is called the Expert Authorizations System, or EASy. EASy processes from 4-6% of all transactions handled by Equifax Check Services, depending on the settings of certain business parameters. On-line transaction-based systems are called directly by retail point-of-sale terminals for the majority of decisions; most of these decisions are comparatively simple to make and usually result in immediate check approvals. Suspect transactions are referred with a request to contact an authorizations center.

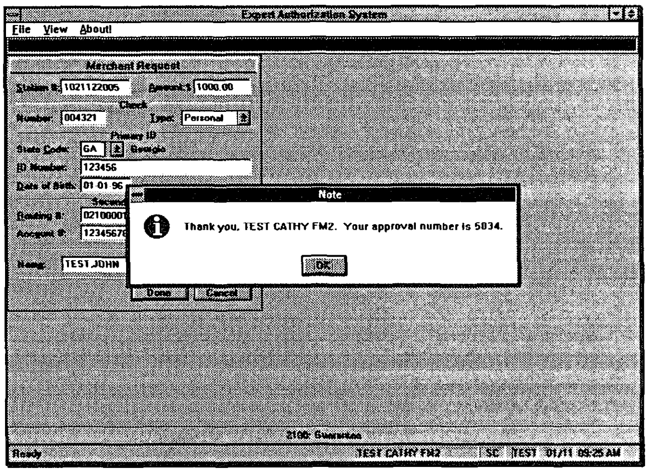

Figure 1: Approval Notification

When the merchant calls in to an authorizations center, an authorizations agent using EASy retrieves certain information about the merchant's customer (the consumer), and the check which caused the referral. This identification and transactional information is processed by EASy in combination with various types of information held at Equifax, and an authorization decision is returned.

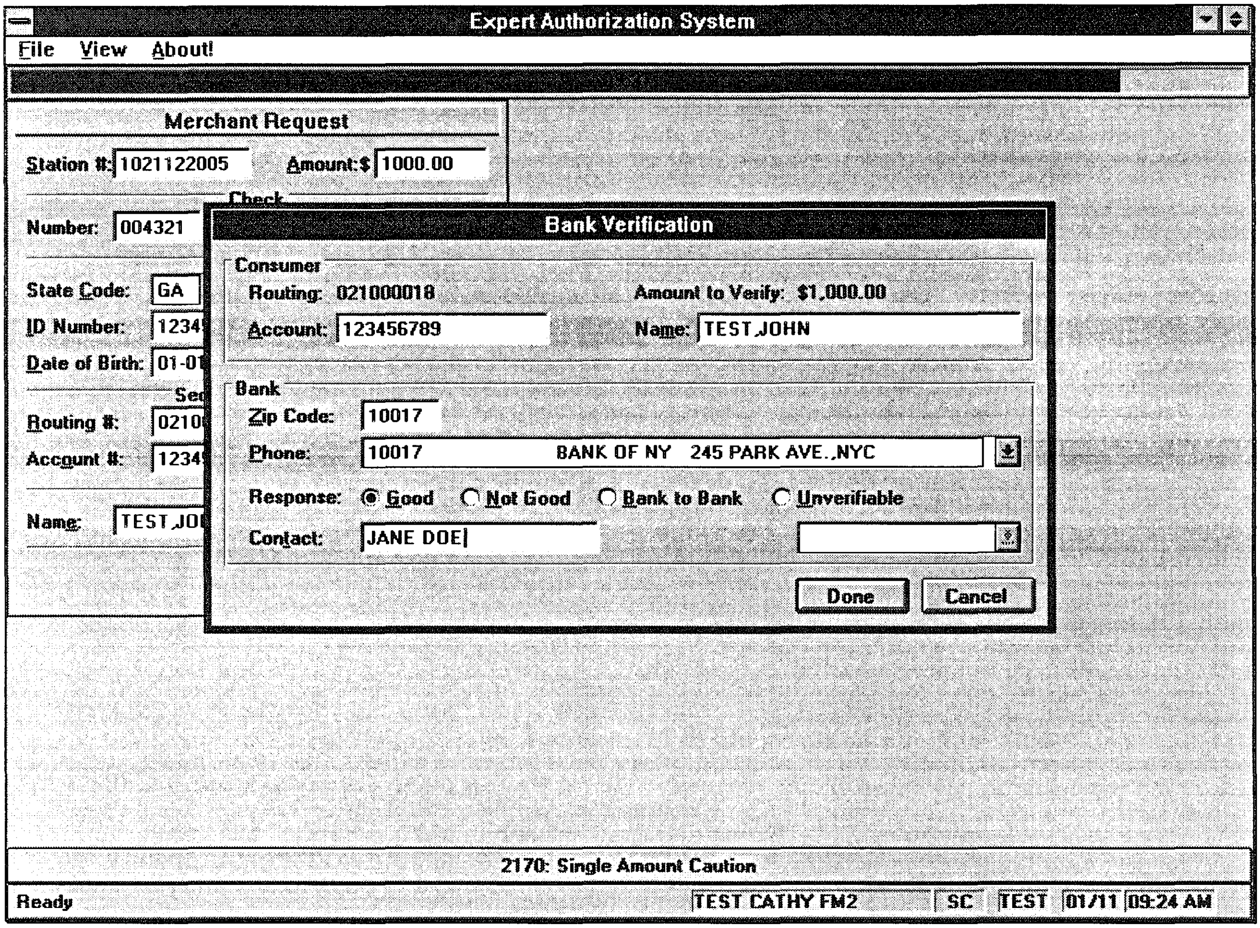

Figure 2 : In-Process Funds Verification

Figures 1-2 provide examples of some of EASy's graphical screens using fictitious data. Basic consumer information is entered by the agent through data request screens similar to that in Figure 1. The types of consumer data, and the order in which it is obtained, is controlled through rule firings in the EASy knowledge base.

As EASy progresses through its decision-making, it may request that the agent obtain additional data or perform certain actions. An example of the latter is shown in Figure 2, where EASy has requested that the agent obtain funds verification from a bank.

Once all the necessary information has been gathered and the processing completed, EASy will display an authorization decision as shown in Figure 2, which shows an example of an authorization approval message. There are a variety of approval, decline, and informational messages that can be displayed depending on the results of the decision process.

How the KB software solution fit these tasks

Both the problem domain and the expert knowledge embodied by the agents both through training and experience lent themselves clearly and immediately to a rule-based system. At a high level, the following activities occur over and over again at each agent's station:

- A new transaction arrives
- The system requests information
- Information is processed through a business model
- More information is obtained as needed
- More processing occurs
- A decision is rendered
- The system cleans up and refreshes itself

The basic function of obtaining data via indicators derived from a business model is a nearly classic description of a backward chaining expert system. The goal is an approve/decline authorization decision, and the system must request information to "prove" one or the other goal. However, reasoning forward from data provided by an agent is a more classic forward-chaining mechanism, thus also indicating an expert system approach. Additionally, because of a long history of developing and refining the business model for check authorizations, Equifax Check Services developed a robust set of "rules" governing what actions the agents should take to make a decision.

To capture these various aspects of authorization decision-making, a blackboard type of expert system architecture was designed, with the knowledge sources consisting of various cells of related rules that embodied the specific aspects of decision-making. The knowledge sources are activated using an innovative twist in the standard blackboard architecture that we refer to as rule phases. Further on we will describe how this differs from a classic ruleset approach.

EASy's User Platform

Each agent sits in front of a 486 or Pentium® PC running Microsoft® Windows® 3.1. Upon system boot-up, Windows starts up automatically, followed by EASy's agent login screen. Upon a validated login, agents can immediately begin taking calls.

These PC's locally run a Microsoft Visual Basic® application, a 32-bit ART**Enterprise*® application which operates on top of the Win32s libraries, a Microsoft Access® database, and a network layer to connect to the Equifax LAN. These applications are described in more detail in the next section.

EASy's Pseudo Three-Tier Environment

In addition to the expert system, EASy includes two additional software components, and represents a near-perfect model of a clear three-tier solution architecture. EASy's architecture differs from the purist definition of three-tier only in that the Graphical User Interface **(GUI)** and the Knowledge Base **(KB)** components sit on the same platform, a PC clone running Windows 3.1. The components are modified and/or maintained by different developers, and isolate areas of functionality, but they reside together and communicate back and forth with each other.

The EASy GUI. EASy's GUI was built using Microsoft Visual Basic 3.0. The Visual Basic GUI component contains neither knowledge nor transactional drivers. The extent of its intelligence is to check for some informational validity (i.e, that a number field actually contains a number). For its actions, it depends entirely on the EASy knowledge base.

The GUI utilizes static data stored in local databases implemented through Microsoft Access. Since an important quality parameter for authorization decisions is the average call time, the GUI also tracks and prominently displays call time durations.

The EASy KB. This {roughly 600-rule, 800-function} system was implemented using ART**Enterprise* from Brightware, Inc. All of the business knowledge which Equifax Check Services has developed over the course of its history about how to process checks resides here. The KB fully drives the GUI component, waits for data from the GUI, processes decisions or partial decisions, and then informs the GUI of either of those decisions or directs the GUI to display screens where the agent can enter additional data.

The EASy Database Layer. EASy makes use of numerous sources of data in determining consumer risk. This data resides within various databases maintained by Equifax Check Services. These operate or exist on numerous server platforms, including UNIX workstations and an IBM Mainframe system. Some of these servers are locally-based to the authorizations LAN, and some are remote. Concurrent on-line remote connections supporting Check Authorizations as well as other systems fully support all the required bandwidth problems posed by EASy.

Integrating EASy's Three Primary Components

Visual Basic **(VB)** integration with ART**Enterprise* was accomplished with functionality built into both the GUI and the KB components. On the KB component, functionality was designed to send either messages or requests to the GUI, at different times, for different purposes. The Log window shown in Figure 5 displays these flow-control instructional messages.

On the GUI component, a rudimentary token parser was built on top of a DDE server. Interestingly, the author of the GUI component wanted to improve maintenance issues and enhance communication across the DDE link. He therefore designed the VB DDE-Server's token parser to process a limited grammar modeled after ART*Script, the flexible scripting language within ART**Enterprise*. Thus, ART*Script could send DDE messages with any of a specified set of ART*Script commands, and they would be processed in the GUI component as though VB were an extension of ART**Enterprise*.[1]

In order to integrate the KB component with the Database component, specialized Equifax-proprietary

[1] ART**Enterprise* does contain an integrated platform-independent object-oriented GUI development tool. In mid-1993, Equifax was a beta-tester of ART**Enterprise* while they were moving forward with EASy's design. The EASy team needed to commit to a more functional and extensible GUI for EASy's Windows-based user platforms, where no portability was required. Visual Basic offered an immediate non-beta solution, and was readily integrated with ART**Enterprise*.

network message-passing calls were utilized to send information from the KB to multiple sources, and to request processing or information from any of multiple databases or external application processors. EASy's developers implemented this capability in Borland C®, using a Windows Dynamically Linked Library **(DLL)**, to which the KB would send calls.

Processing Residing Outside the KB

EASy relies on some processing which occurs on external application processors. In these cases, it sends a custom network request to return results for a particular job. Thus, it could be said that some of the application's knowledge resides outside the knowledge base. For example, EASy relies on external processing for the validation and analysis of consumer identification.

Additionally, EASy relies on external applications residing at various computers on the Check Services authorization Local Area Network [See Figure 3]. These external applications provide additional input to EASy regarding a variety of parameters that may affect the authorization decision. These parameters generally involve basic guidelines that are followed in an authorization decision and are derived from a combination of Equifax Check Services' business rules and agreements with merchants utilizing authorization services. These parameters and processes are external to EASy primarily because they are also used for the automatic electronic authorization processing.

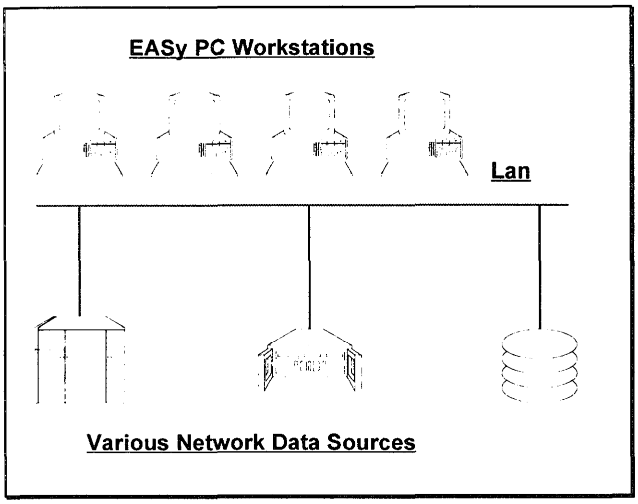

Figure 3: EASy LAN Topology

The types of information returned by these external processes provides EASy with initial information and status indicators for the current authorization transaction. These status indicators are prioritized by EASy and used as a basis for determining the order of processing of EASy's rule phases.

KB Design

Figure 4 illustrates the flow of the EASy KB process. The overriding structure of the KB is a traditional blackboard architecture. In order to organize the decision process the multiple mechanisms involved in the decision process were separated and implemented as independent sets of rules. Traditionally, using ART**Enterprise*, a grouped set of rules is linked using the ruleset mechanism. However, the ruleset mechanism was felt to be too restrictive for the EASy application for several reasons, including:

1. Check Authorizations management felt that custom processing options at a customer-specific level was a necessary future enhancement for EASy in order to deliver all proposed service enhancements; this was always envisioned as utilizing rule-set capabilities. It was determined that all possible custom rules for a given client would not easily be classified into singular areas of the knowledge base, and therefore should not be implemented as rule parameters.
2. It became obvious that different actions could occur within a given processing area, under the same conditions, depending on what might have occurred in the knowledge base beforehand. So the knowledge base at a micro level was non-deterministic, but was still deterministic at a macro level. Investigating further, two classifications could be made for particular rule-groups within the knowledge base. There were those that could be "called" at the *top level*, and those that could be "called" as *functions* of higher level knowledge base areas. The sequencing of certain rule-groups was critical to the decision process.

Based on the above reasons, rule groups became *phases* of a transaction. A deterministic set of all paths through these phases in any given transaction was mapped out, and mechanisms were created to allow orderly flow of a transaction through its phases. When rules and their corresponding exceptions needed to be able to fire under several different phases, their host phases could be enabled concurrently with other phases.

Like rulesets, phases were implemented with a simple control fact mechanism, but additional facts would be asserted in parallel. These would indicate which phase led to the current phase, how the current phase had been entered, and if the current phase was completed, indicating that control could be passed to another phase. This can not be fully viewed as procedural, because multiple phases can be concurrently enabled under known circumstances.

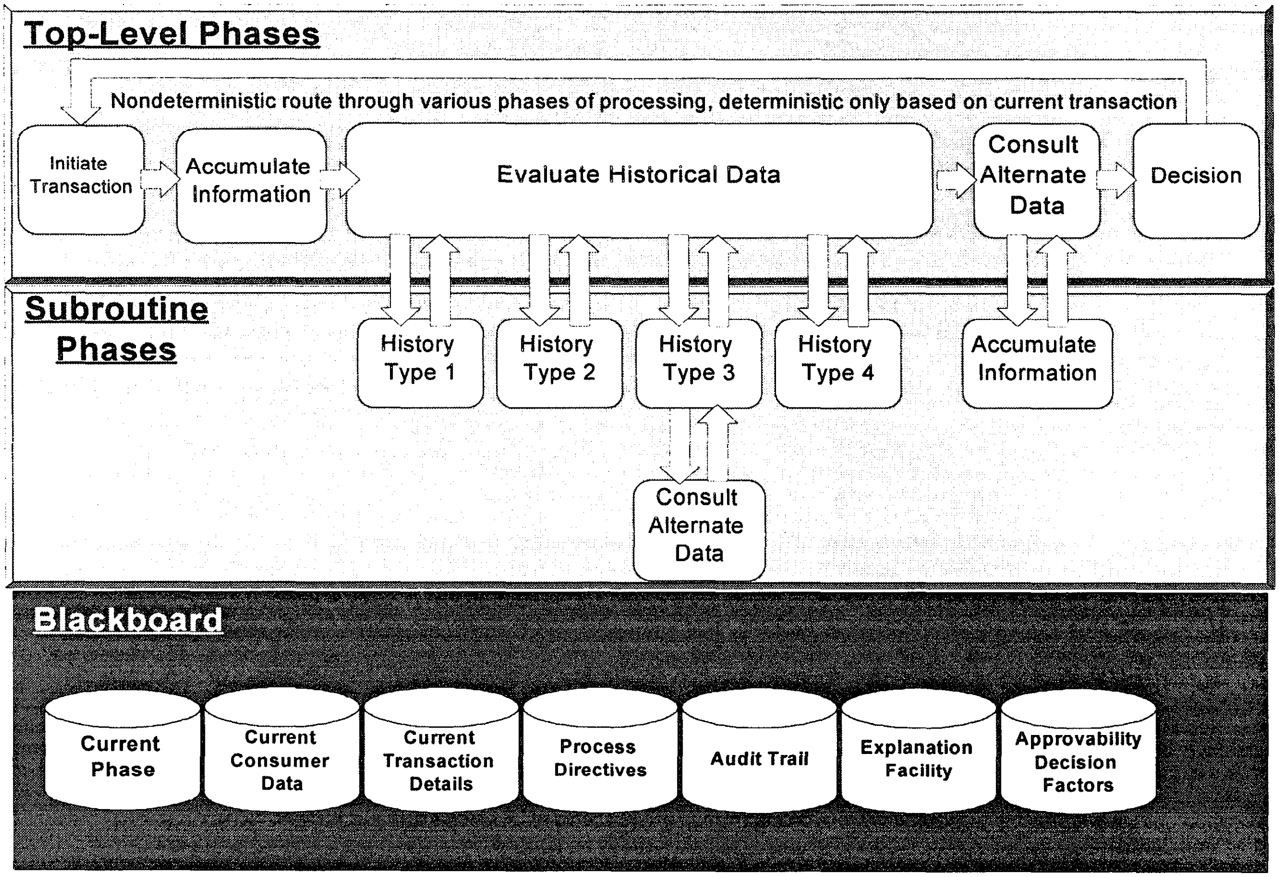

Figure 4: Phase Topology and Blackboard Architecture

Within each phase of processing, the EASy KB primarily uses forward-chaining, data-driven rules to accomplish its decisions in a classic non-deterministic fashion. However, as previously noted, one of the major features of the EASy KB is its ability to drive a user-interface and obtain additional data as needed, which is usually a function of backward-chaining rules.

The backward-chaining mechanism was directly implemented using a few simple rules that fired based on facts asserted within the forward-chaining phases. These facts were essentially goal facts which specified that certain types of information should be transferred to the GUI, and that the KB should wait for a response from the GUI. These GUI rules have the lowest priority in the KB. The KB behavior, in effect, is to process as much as it can on the current information, assert GUI goal facts if a rule determines that more information is needed, and then let the forward-chaining phases essentially run out of rule activations and allow the GUI rules to fire. The GUI then asserts the obtained data back into the KB, which in turn causes the forward-chaining, data-driven mechanism to resume.

Phases Mechanism and Topology

The following diagram illustrates the flow of phases and their relationship to the KB Blackboard.

Please note that in the following rule examples, all proprietary details have been left out.

A Phase Switch Initiation

```
(define-rule Switch-to-Phase-A
  "Recognize the switch moment for phase A"
  (phase current-phase & ~A)
  (transaction data one)
  (transaction data two)
  ...
=>
  (assert (switch-to phase A)
          (completed-phase ?current-phase))
)
```

In this generically defined rule, a firing would indicate that Phase A must be scheduled. It watches working memory to insure identify what phase is currently active. If the phase it wants to schedule (in this case A) can follow the currently active phase, and other conditions for phase entry are met, then the request to switch to the new phase is submitted by fact assertion. The Blackboard phase transition rules then retract the current phase gating fact [e.g. `(phase current-phase <phase-name>)`], and assert the necessary activation facts for the new phase. In a phase switch action, the current phase is considered completed and the new phase is considered active.

A Blackboard Phase-Switch rule

```
(define-rule PHASES:PHASE-SWITCH
  "Effect the transfer of phases"
  ?f1 <- (phase current-phase ?)
  ?f2 <- (phase switch-to     ?next)
  (phase-stack) ;empty stack, toplevel only
=>
  (retract ?f1 ?f2)
  (assert (phase current-phase ?next))
)
```

A follow-up rule-firing to the previously described example would be the standard Phase-Switch rule, shown immediately above, which effects all switches at the toplevel, regardless of their name/purpose. It is treated as being a Blackboard rule due to its applicability at any time throughout a transaction to any pair of phases.

An Intra-Phase Concluding Rule

```
(define-rule Sub-Call-Phase-A
  "Recognize the subroutine moment for A"
  (phase current-phase & ~A)
  (phase-stack $?SEQ
    &:(not (member$ A $?SEQ))) ;prevent loop
  (transaction data one)
  (transaction data two)
  ...
=>
  (assert (sub-call phase A))
)
```

In this rule, a firing would indicate that EASy should suspend the current phase for later re-activation, and enter Phase A much like a procedural subroutine. A decision from Phase A is required to assist the current phase with its own decision making. Note that a fact relation `(phase-stack...)` is maintained on the Blackboard, in order to store an ordered list of the current phase sub-calls. Localized facts used in the higher phases remain unused, with no matches, until their appropriate phase scope is re-activated.

Application Use and Payoff

Improvements in Agent Training

EASy has been fully deployed and in operation since February of 1995. During off-peak season, the minimum number of concurrent daytime users at all authorization sites is approximately 100. Given that the system operates 24 hours a day, 7 days a week, the total number of Authorizations Agent employees is higher.

During the height of peak season, the total number of transactions received by the on-line authorization system has reached 800,000 per day. On a peak day, up to 300 concurrent EASy users have taken 45,000 calls per day, where each call represents a transaction that has been referred from the on-line authorization system. Since the holiday shopping process ramps up substantially earlier than Thanksgiving, it takes time to ramp up the number of authorizations agents; training for peak used to be very costly.

As much as six weeks of training for each employee, using a methodology which required substantial individual attention, has been reduced to 3 days of group classes and supervision. As a result, EASy has reduced total training time by about *ninety per-cent* per year. This dramatic reduction in time, and therefore cost, has enabled EASy supervisory staff to more easily fill any openings caused by employee turnover during the peak season, and thereby maintain high levels of service throughout peak.

A residual but significant benefit of reduced training time is that agents can begin taking calls far earlier than before. This requires less training schedule management than before, because under EASy, extensive training courses do not need to be staggered over long periods of time. This directly and rapidly increases the productivity and consistency of decision-making in the authorizations department.

Enabling of Wider Service Offerings

Certainly an important future benefit is the enabling of customer-specific service offerings. The flexibility of the system provides for the incorporation of these specialized single-customer rule-sets in the short term. Work toward this area has already begun.

General System-Wide Benefits

Through the use of EASy, the time it takes to provide an authorizations decision has been slightly reduced, with significantly increased levels of confidence and reliability. However, EASy provides much more functionality than was available in the legacy system, and therefore some of the time savings of a KB implementation is given up to new processing and increased intelligence. This directly reduces the number of agents required during peak hours while providing shorter wait-times.

To help agents reduce their call times, they now have a color-coded timer bar running across the top of the Visual Basic GUI to indicate how much time has elapsed. At specific timer intervals the color changes from Green, through yellow and finally to Red. If a call that has extended into the red timer bar has not been resolved after a short while, the red timer bar begins flashing to provide additional impetus to complete the call.

Because of the reliability of EASy's standardized decision making, Equifax no longer must expend resources to transaction-process-monitor (TP Monitor) Check Authorizations that get routed to the authorizations center. Once the expert system has been validated by the experts and distributed to all the EASy stations, it needs no on-line monitoring. Should any issues arise, agents are able to report any problems or anomalies through their supervisory staff.

Application History

EASy Project History

EASy was designed and implemented to the point of deployment within 18 months, and has been deployed and fully operational since February of 1995.

EASy was implemented in stages, using an experimental approach to identify the best method of storing information, and the most optimal mechanisms for rule pattern-matching. Purity of design was originally sacrificed in order to provide proof of concept through rapid prototyping. This was done in order to provide continuous feedback as to the value and productivity of development within the rule-based paradigm.

The rapid-prototyping approach was combined with a modified spiral methodology, thereby providing for the effective inclusion of new developers on the project at any stage.

In practice, as EASy's multiple components approached completion, and testing/validation efforts rapidly increased, a waterfall methodology took precedence in order to validate new functionality and bug-fixes.

The following describes the major steps in EASy's development. The # of PC's refers to the total install base in a given time period, which is a superset of the total number of concurrent users at any given moment.

- **July 1993 - July 1994:** *Primary development effort. Resulted in 8 PC's running an early test version periodically under structured testing against real calls.*
    - *5 developers total, 3 at any one time.*
    - *0 Users.*
- **July 1994 - September 1994:** *Bug Fixing combined with an upgrade from ART*Enterprise 1.0.Beta to ART*Enterprise 1.0.General Availability. Resulted in 16 PC's running a newer test release of EASy, still under strict supervision but more frequently than before.*
    - *5 developers total. 3 at any one time.*
    - *0-16 Users, Alpha Release.*
- **October 1994 - December 1994:** *Calls were being taken on a regular basis on 16 PC's with a more stable and functional version. Development during this phase included some performance tuning, application re-engineering, and addition of new functionality.*
    - *4 developers total, 3 at any one time.*
    - *16-50 Users, Beta Release*
- **December 1994 - February 1995:** *A completion and cleanup of re-engineering effort; result is a new phased blackboard software architecture. and a new EASy which is deployed on 80 PC's locally, and 50 PC's at other locations.*
    - *3 developers concurrently.*
    - *50-130 Users, Initial Full Roll-out.*
- **February 1995 - April 1995:** *A concentrated period of enhancements and system stabilizing. Resulted in an unprecedented level of confidence in decision making. Peripheral results included the elimination of multiple extraneous KB mechanisms which had been built over time as patches, and not yet removed as part of the new phased architecture.*
    - *2 developers concurrently.*
    - *Full user deployment based on seasonal requirements.*
- **April 1995 - September 1995:** *Virtually eliminated reliance on legacy systems by adding to EASy the capability to interface with a new PC-based Authorizations System which provides suggestions for how to process transactions.*
    - *2 developers concurrently.*
    - *Full user deployment based on seasonal requirements.*
- **September 1995 - November 1995:** *During this period, developers and testers performed a "pre-season shake-out" in order to hammer out any rare but potential problems before the peak season arrived.*
    - *2 developers concurrently.*
    - *Full user deployment based on seasonal requirements.*

- **November 1995 - Present:** *Stable Operation, minimal risk. Equifax Management approval has been obtained to initiate an extensive feature-enhancement program.*
    - *2 developers concurrently.*
    - *Full user deployment based on seasonal requirements.*

How was EASy validated?

A parallel testbed system, using the same hardware as the production system but entirely different data paths, was developed to mirror the same capabilities available in the production system. Sample transactions could be run and recorded against the testbed system without any impact to production data or actual consumer transactions. The experts consulted by the Knowledge Engineers were the authorizations management and supervisory staff, which comprised the primary testing and validating group.

As either new functionality or bug fixes were coded, an update to a set of test PC's would occur. The experts would run real transactions which were known to be standard and/or boundary cases against the testbed EASy. If the system passed all the obvious tests designed to weed out problems, unstructured regression testing would occur. A set of standard unrelated transactions would be run to validate that existing functionality had not been broken. Those transactions had not been formalized largely because the experts could type in those transactions faster than developers could write them down.

Should any problems have missed detection by the experts, they were certainly found in production. Calls would be routed for particular problems to an agent who would complete the transaction manually, and a knowledge engineer could sit at that agent's PC, interrupt the production transaction, and call up ART*Enterprise's Command Interpreter to investigate the state of the problem. Usually the developer would immediately be able

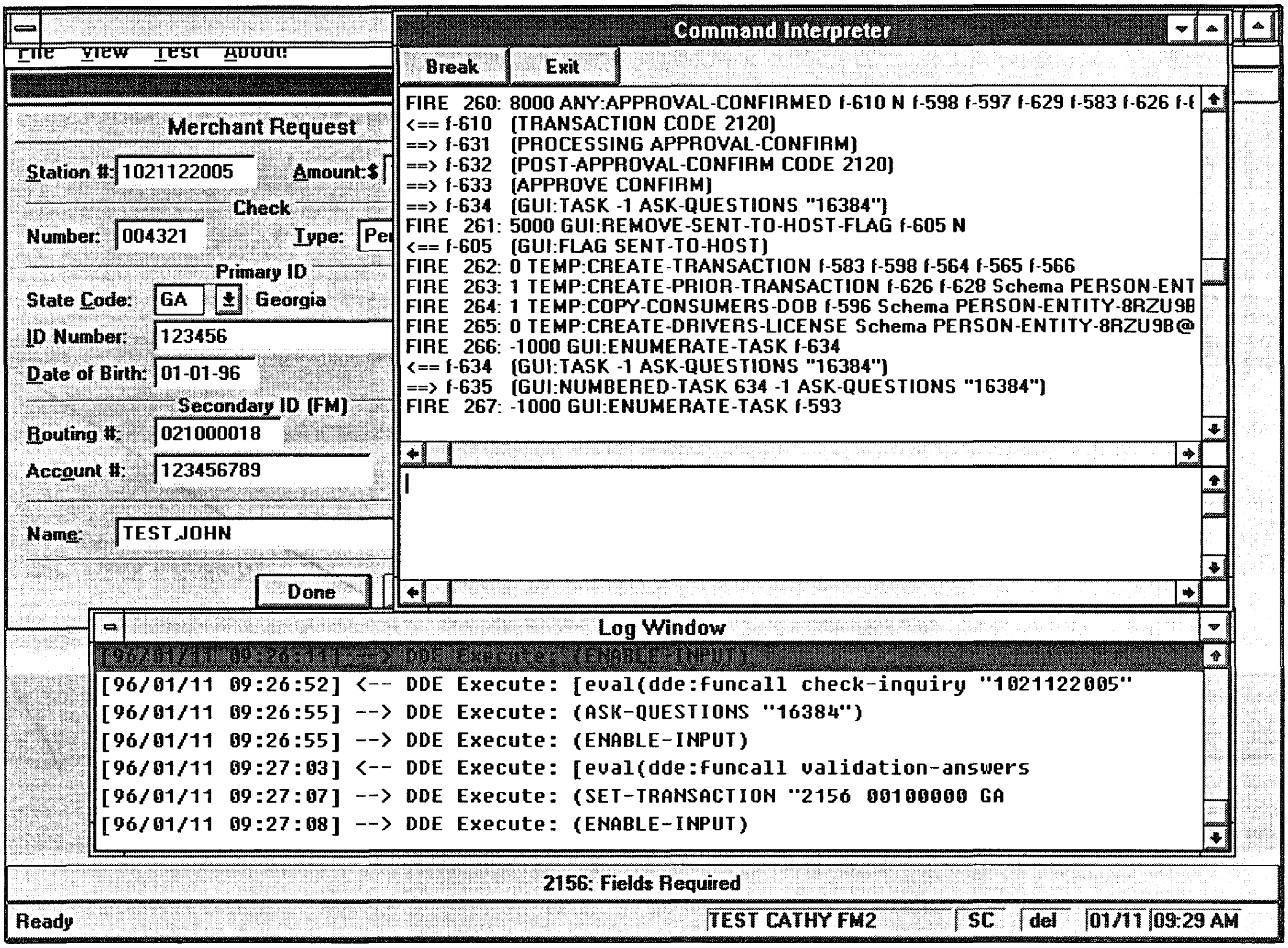

Figure 5: KB Development Environment

to tell what went wrong in the knowledge base or GUI. However, the transaction would always be completed such that the merchant received an appropriate decision, even if this required supervisory intervention.

ART*Enterprise provides a highly customizable development environment. To simplify the development and maintenance of EASy, a specialized development environment was created by unlinking the GUI and other unused ART*Enterprise tools, and adding functions and browsers that made rapid development and debugging of EASy code faster and easier. Figure 5 shows an example of a debugging session using a custom DDE message log window, and an enhanced Rule tracing facility.

With the modified bare KB studio, developers rapidly developed command-line functionality and skills for debugging EASy that allowed for rapid problem isolation.

Coupled with the visually minimal but significantly extended ART*Enterprise environment, a special DDE Message log-window provided through the EASy GUI, enabled only for developers, makes for a very ideal problem-solving environment in the EASy application.

The EASy Deployment Process

It took six months to deploy the EASy system on all agents' stations as a replacement for their Tandem screens, from July, 1994 to January 1995.

To elaborate on the itemized timeline shown 2 sections prior, at first, 8 EASy PC's were brought up initially for testing purposes. Shortly after 8 seemed stable, 8 more were installed, but these were still primarily for structured periods of monitored testing. After 2 months of testing and debugging, EASy had stabilized to the point of leaving those 16 stations up and running continuously. By peak time (early-mid November, 1994) the number of EASy PC's had grown to about 50 workstations.

A policy had been issued by Equifax management that no new EASy releases could be installed during the peak season. This offered a prime opportunity for the developers to begin thinking about how to re-engineer the knowledge base. Equifax needed to be convinced that the investment in that activity would have clear payoffs. The knowledge engineers had found that they had reached a point of diminishing returns on working on bug-fixes. Thus spending time on debugging seemed to be the more questionable investment, as opposed to imposing a new data flow architecture on existing mechanisms.

Based on these and other reasons, Equifax gave the go ahead to complete the new knowledge-base architecture.

Before peak season was over (last week in December 1994), that re-engineering had begun, and by the middle of January, a re-engineered EASy was fully deployed across all the authorizations agents.

In the opinions of the authors of this paper, three elements contributed to the rapid completion and success of EASy's modification.

- The use of a knowledge-based paradigm had created within EASy the cellular groupings of data processing that automatically lent themselves to isolated rule-firing chains. Mechanisms that interoperated between these cells were easy to distinguish, clean up, and structure, such as control facts, absence of control facts, objects, and various types of consumer and merchant data.
- The use of ART*Enterprise as the Expert System development tool, which provided the ability to extend development functionality, eliminate unrelated tools from the Studio, and provide run-time debugging capability. ART*Enterprise/Windows allowed for integration of C code both directly and through DLL's, as well as setting up and maintaining DDE communication with the GUI component of the application.
- The creation of a team of knowledge engineers, GUI programmers, expert users and committed managers, all of whom provided Equifax with an unsurpassed level of aggregate knowledge, creativity, skill, and commitment to improving quality and supporting new technologies.

Application Maintenance

How EASy is Maintained and Updated

EASy is a living system which must comply with new and updated legal requirements affecting financial risk and authorizations systems. As a result, the system is never considered fully completed, and must be updated regularly. Knowledge Engineers are on staff at Equifax to update the EASy knowledge base, and they divide their time amongst updates, fixing bugs, adding functionality, and performing unit testing.

To achieve uniformity with respect to new system releases, Equifax's Quality Assurance manager maintains revision control and system update distribution. In order to release a new version, the new release must pass a variable suite of basic transactions, and be fully tested with boundary conditions against any bugs found through monitoring particular transactions against the previous release. Regression testing was not formalized due to the variable growth of the system over time under a rapid prototyping approach.

No hands-on distribution is necessary. To release a new version that has passed all compilation and testing steps, the QA manager sends the new software to a location on the LAN, and informs the supervisory staff for Authorizations that a special utility program for installing new versions of EASy can be run on each EASy station. This utility reconciles the current versions of all EASy applications and files with those found on the local PC.

The entire update process can be accomplished in less than an hour, assuming no errors were found during compiling and testing.

Who maintains EASy and how often?

Two full-time programmers/knowledge engineers maintain the system currently, alongside efforts to add/modify functionality. The users of the system are responsible for identifying any problems with the system, and their supervisors are responsible for determining what constitutes a user error vs. an actual system problem. Code modifications are made by knowledge engineers, tested against a testbed system, and agent supervisors then test the system in order to approve the code modifications prior to installation in the production environment.

Modifications to the system now occur once every 2 weeks, and these now encompass almost entirely new pieces of functionality, as opposed to bug-fixes.

Does EASy know more over time?

EASy must comply with changing state/local laws and Federal industry regulations, and so must be modified periodically to address such legal and regulatory concerns. Further, EASy has enabled the addition of new business knowledge that would have been difficult or impossible to implement with the legacy mainframe/Tandem based solution.

Does EASy's design ease/enable modification?

Several issues contribute to EASy being easily modifiable, including the choice of AI technology in general, the choice of ART**Enterprise* as the tool to provide that technology, and the choice of a phased blackboard architecture.

It was a central deliverable of the application to be able to change over time. The extent to which the system could be modified was proven during the phased architecture re-engineering process. In few other paradigms than a rule-based approach can you take a complete existing system with little structure and impose a clear structure in just over a month, automatically eliminating a significant number of bugs, without causing additional ones. In a procedural paradigm a rewrite would likely be required for the same level of modifications.

Future Plans

The current phasing architecture has provided EASy with a good deal of flexibility in dealing with the process of authorization decision making. The current phasing methodology is based on an underlying business model which could be more flexible in terms of what phases can fire and in what order. A fully non-deterministic approach would improve the ability to allow more variable authorization processing. To accommodate these requirements, the authorization business model is being sub-divided into its component processes. When this has been completed, an improved methodology for selecting custom authorization processing may be possible.

As might be imagined, this presents a variety of challenges for the current phase-based blackboard architecture. Although phases may be clearly considered as components, in hindsight, there are alternative ways to view a component-based authorizations decision which may be more applicable. For example, a specific customer may want components A, B, and D to be applied to their authorizations, skipping component C. This could present problems for both components B and D if component B, when viewed as a phase, wants to "naturally" transition to a phase (component) C. Also, component (phase) D may naturally require information normally supplied by component (phase) C. In the above example, a component may be required to process a transaction with incomplete information. A phasing architecture in this case may be required to schedule phases in a partially deterministic fashion based on customer requirements.

Work on these modifications to the EASy phase-based blackboard architecture has already begun. Equifax is re-examining the current business model to determine the extent to which partial information can be effectively applied in a component-based system, in order to reach a very clear (i.e. not partial) approval decision. As a result, the granularity of the components is also under review.

Summary

Through the Expert Authorizations System, Equifax Check Services has improved the quality of their existing services, and significantly enhanced their current and future services offerings. All of these benefits are now provided at lower cost to Equifax than the previous Tandem-based system could have allowed.

Since Equifax has shown EASy to numerous other departments, Expert System technology has received a high-level of buy-in and visibility. Other internal groups have begun exploring how to incorporate innovative AI approaches to improve their own productivity and quality.

In addition, through improved training quality, reduced training time and costs, and the user-friendly EASy environment, the authorizations department can now provide improvements in customer satisfaction.

Acknowledgments

The authors of this document would like to extend their hearty thanks to the following individuals who have been involved with EASy, in varying degrees, over the course of the project:

From Equifax: Greg Mallare, Margaret Fortson, Bill Overbay, Tim Prosser, Mike Hernandez, Cathy Reed, Lynda Patry, John Storch, and many others who have provided indirect but equally important assistance.

From Brightware: Bill Richer, Jeff Livesay, Kate Murphy, Greg Hadaller, Raj Rao

Special thanks go from Brightware to Greg Mallare and Margaret Fortson of Equifax, who were committed beta-testers and users of ART**Enterprise*/Windows from the earliest releases onward.

Trademark Acknowledgments

References

Dzierzanowski, J., Chrisman, K., MacKinnon, G., Klahr, P., 1989. The Authorizer's Assistant, A Knowledge-Based Credit Authorization System for American Express. *Proceedings of the 1989 Conference on Innovative Applications of Artificial Intelligence*, AAAI Press, Stanford, CA.