EXSYS® CORVID™ Java-Based Expert System Knowledge Automation Development & Deployment Technologies

WHITE PAPER

Contents

Exsys CORVID - The New Approach

Exsys CORVID is a revolutionary approach to developing and fielding online Knowledge Automation Expert Systems - for the Web, and as stand-alone systems, which can be delivered cross-platform. For the first time, it is possible to convert complex problem-solving logic into a form that can be easily incorporated into a Web site. Providing access to expert decision-making knowledge is the best way for businesses to attract and retain customers, realize efficiency gains, and have a significant competitive edge.

For over 20 years EXSYS Inc. has been the leader in providing proven technology to build and deploy interactive knowledge automation expert systems, which deliver a company's most valuable asset, expert knowledge – to the people that need it, whenever they need it. Distributed enterprise-wide, organizations are increasing productivity, cutting costs and improving customer relations – while creating new profit centers and achieving demonstrable return on investment.

Exsys CORVID converts expert knowledge into a form that allows delivery of advice and recommendations to improve performance, capability and efficiency, while reducing training and costly errors. EXSYS products are used by over 50% of the Fortune 100 companies, many Government agencies, and thousands of businesses and organizations.

CORVID was developed by EXSYS Inc. - the company that first brought practical expert system development to the PC in 1983, and first brought expert systems to the Web in 1996. EXSYS Inc's founder, Dustin Huntington, and the company's software developers have over 21 years experience in the artificial intelligence field and in designing practical expert system development tools. The original *EXSYS Developer* software is the longest lived such product on the market, with continuous use in business and industry for over 20 years.

Exsys CORVID resulted from a broad and intensive examination of what is needed to build and field expert systems in today's Internet oriented world. CORVID is designed to provide a unique powerful product that allows businesses and organizations to easily develop and field powerful online advisory applications, while still maintaining EXSYS's long support for other types of expert systems.

Feature Benefits

CORVID brings 3 revolutionary changes to knowledge automation expert system development.

- **Understandable, Systematic Rule-Based Logic**
- **A Practical, Efficient Development Environment**
- **Easy, Cross-Platform Java-Based Web Deployment**

In examining how to enable users to most effectively build expert systems, many approaches were considered. EXSYS products have traditionally used a pure **rule-based** approach to logic and this has proven to be extremely effective for thousands of users.

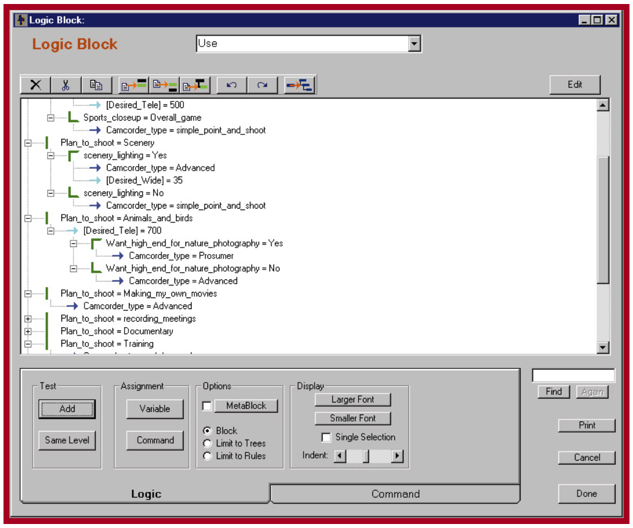
Some other expert system tools have taken a rigorously **object-oriented** approach, which results in a much more difficult and complex development environment. An object-oriented approach is very effective for programming, but often does not work well for human decision-making processes. Most decisions are thought of and described by people as, "If…. Then…" type logic. Often this does not easily convert to a class hierarchy or an object-oriented approach, and does not match the way they think about the decision-making process.

While many decision-making problems are not easy to express in an object-oriented methodology, there are tremendous benefits to using **object-structured** components. This difference between true object-oriented programming and object-structured programming is what has made Microsoft's Visual Basic (VB) so popular. VB is not a true object-oriented programming language, though it provides many object-based features. This simplification has allowed VB to be used by more programmers than practically all other languages combined.

Exsys CORVID is based on the VB model. It provides an *object structure* that makes it easy to build systems using methods and properties of Variable classes, while not requiring the developer to change the way they think and describe their decision-making steps and logic. The result is a very flexible and powerful development environment that can easily be learned.

CORVID has 7 types of Variables that provide a wide range of functions. The powerful Collection Variable allows data to be "collected" from multiple runs and enables many new capabilities. A variety of methods and properties for all of the Variables allow them to be used in new and unique ways to solve problems and display results.

CORVID Logic Blocks



Logic Block:

Logic Block

Use

```
        [Desired_Tele] = 500
    Sports_closeup = Overall_game
        Camcorder_type = simple_point_and_shoot
Plan_to_shoot = Scenery
    scenery_lighting = Yes
        Camcorder_type = Advanced
        [Desired_Wide] = 35
    scenery_lighting = No
        Camcorder_type = simple_point_and_shoot
Plan_to_shoot = Animals_and_birds
    [Desired_Tele] = 700
        Want_high_end_for_nature_photography = Yes
            Camcorder_type = Prosumer
        Want_high_end_for_nature_photography = No
            Camcorder_type = Advanced
Plan_to_shoot = Making_my_own_movies
    Camcorder_type = Advanced
Plan_to_shoot = recording_meetings
Plan_to_shoot = Documentary
Plan_to_shoot = Training
```

Options: ☐ MetaBlock; ● Block ○ Limit to Trees ○ Limit to Rules

Display: ☐ Single Selection

Logic Block Window

CORVID introduces a new concept in managing decision-making logic – **Logic Blocks**. In previous EXSYS products, there are tree logic diagrams, and individual If/Then rules. Many systems required multiple trees and rules to collectively provide each decision-making step, with no consistent way to organize the related logic. CORVID's Logic Blocks are a superset of rules and trees, with some new features not found in either. A Logic Block can be any combination of rules and trees that have a related function. This allows the logic to be organized into blocks that behave as objects. A Logic Block can be anything from an entire knowledge base to a single rule – it all depends on what the problem calls for.

Logic Blocks can be run using forward or backward chaining. They can also be associated with a spreadsheet file to apply the logic in the block sequentially to each row. This enables a **product selection** system to be built that has all the "generic" logic in blocks, and all the product details in a spreadsheet file. Updating and maintaining the system is as simple as changing the spreadsheet to have new data. Product selection problems have never been simpler – especially for situations where the product line or details are frequently changing, such as inventory or pricing.

Logic Blocks are built and maintained in a very visually oriented development environment that is easy to learn and use. The underlying knowledge representation in the block is still an If/Then rule, making it easy to read and understand, but a Logic Block provides a way to organize and use the rules in a way not previously possible.

Fast Web-Enablement Through Java Delivery

The Web offers the ideal way to reach people. It is the gateway to the new economy and the primary conduit to prospects, customers, partners and employees. Exsys CORVID offers a revolutionary way to provide interactive expert systems on Web sites. This can bring top-level advisory sales and support expertise to the forefront to help site visitors make quick, correct and consistent decisions. CORVID systems can be delivered over the Web two ways - a servlet that runs on your server and generates HTML pages to ask questions and present results, or an applet that is simply added to a Web page and provides the full interactive system with all processing done on the client machine. The same system can be delivered in either or both delivery modes to provide situation specific answers and recommendations that your users need.

The **Exsys CORVID Servlet Runtime** is a Java servlet that can be run with Tomcat or a comparable servlet engine. It implements the proven CORVID Inference Engine to process the logic of a system, determine what questions to ask the user, analyze the user's input and arrive at conclusions or recommendations.

HTML templates define the look-and-feel of the end user interface to match a Web site. The full power of HTML, and any extensions such as XML, Java Script, etc. can be used to create sophisticated user interfaces. Sample templates are provided with CORVID making it easy to field your systems. The templates can be generated and edited with any HTML editor to match your Web site design.

Systems can be integrated with personalization software allowing it to automatically provide some of the data the system will need. Built in database access and report generation capabilities allow you to obtaining data from your existing databases and store results.

Since only the HTML page is sent to the user, interaction is fast and can even be done with palmtop browsers and other wireless devices. The Servlet Runtime allows cookies to be used, but does not require them. This allows it to be integrated with other programs, while still performing correctly for users that have cookies disabled on their browsers.

CORVID also offers an alternative way to deliver Web-based systems via a Java applet. Other companies have tried Java applet runtimes, but they were generally very large and slow to download. Often they had very limited interface design options, or required that the system developer also be a Java programmer. Exsys CORVID applications are delivered by a small (~130k) applet, which provides robust interface design options. The system author just selects how a question is to be asked, and what text or graphics to associate with it. If multiple questions need to be asked together, the author just makes a list and it will happen. Questions can be asked through a variety of controls and styles, and can also include image maps.

Since the system is running as an applet, HTML can be used for the rest of the page surrounding the applet. CORVID supports many ways to add links to other pages from graphics and text, with the links displayed in separate browser windows or frames on the same page. When the system reaches its conclusion, even more flexibly is available in how results are displayed. The combination of graphics and text makes it easy to build systems with a sophisticated look and feel, which can be very rapidly integrated into existing Web sites.

The CORVID Runtime options provide all the functionality needed for most systems. In addition, if special system features are required, the CORVID Runtime can communicate with other programs for additional functions. Exsys CORVID can work with external sources of data, or content managers associated with a database, to pass data into the Runtime on startup. For data that is only available or best calculated on the server, CORVID has many ways to perform server-side functions including CGI, ASP and ISP. For an even higher level of customization, user-defined functions and interfaces can be added to the CORVID Runtime itself through the CORVID API.

For expert systems where delivery via the Web is not required, the CORVID Runtime can function as a Java application and run on any platform that supports Java. In this mode, CORVID can access local files, and run either interactively or in a batch mode to process data.

CORVID is a new generation of knowledge automation expert system development product. Whether you are adding expert decision-making knowledge to a Web site or building a stand-alone application that can run on any platform, expert systems have never been easier or more practical.

Exsys CORVID Knowledge Automation Expert System Development

Heuristics and Rules

In expert system terminology, each of the expert's "rules of thumb" is a heuristic. That is a specific small fact that tells how to make a part of a decision. The combination of all the heuristics allows the overall decision-making problem to be solved. In our brain, we combine these individual heuristics intuitively and systematically. You don't have to stop and say, "now I need to know *this* to help make that decision…", our brain just does it.

A large part of building an expert system is identifying the individual decision steps and converting them into a form that a computer can use.

There are many ways of describing the heuristics for a decision-making process, but the one that has proven the most effective and efficient is the **IF/THEN rule**. This is a rule where there is an IF part that can be tested to be true or false based on the data for a specific case or situation. When the IF part is true, the statements in the THEN part are also considered true. This is how a basic rule is written.

```
IF
        It is raining
THEN
        You should wear a raincoat
```

With Exsys CORVID, these rules are very similar to the form that you would use to explain the heuristic using English and algebra. For example, *"If the investment customer has a high risk tolerance and requires rapid growth to reach their objectives, Mutual Fund X would be a good choice."*

In a rule this would become:

```
IF
        The customer has high-risk tolerance
AND
        Meeting objectives requires rapid growth
THEN
        Mutual Fund X is a good choice
```

This rule includes a small amount of syntax, but it is still very easy to read and understand what it means. If you built similar rules for each of the heuristics in the decision-making process, you would have the logic for the expert system.

The CORVID Inference Engine

Our brain processes and combines the heuristics intuitively. Unfortunately, a computer is nowhere near as effective as our brain. In Exsys CORVID a special program called an "Inference Engine" is used as the "brain" to analyze and combine the individual rules to solve the larger problem.

The Inference Engine determines:

- The possible answers to the problem
- What data is needed to determine if a particular answer is appropriate
- If there is a way to derive or calculate the needed data from other rules
- When enough data is available to eliminate a possible answer, and stop asking unnecessary questions related to it
- How to differentiate between remaining answers
- Which answer is most likely, based on the rules

It is the Inference Engine that makes If/Then rules in an expert system very different from If/Then commands in a computer language such as Visual Basic or C++. Rules are **not** equivalent to lines of code; they are facts that are automatically combined in various ways by the Inference Engine. This makes the expert system approach far more powerful, effective and maintainable for knowledge delivery than traditional programming techniques.

Backward Chaining / Forward Chaining

One way in which the CORVID Inference Engine combines the rules is called **backward chaining**. Backward chaining is "goal driven". Setting appropriate goals is part of the expert system development process. Typically the top-level goals are the possible answers to the problem or potential recommendations. The Inference Engine can determine what information it needs to meet a particular goal, including determining when that goal is met or that a goal is not relevant.

The Inference Engine analyzes what data is needed to determine if a possible goal is appropriate for the user. To make this determination, the system requires data on the specific situation being analyzed. This data can come from other rules, external sources such as databases and spreadsheets, or by asking the user additional questions. Using the previous example rule, suppose the first goal is to determine if the expert system should recommend Mutual Fund X. The Inference Engine checks the rules to find one that would be relevant to making this decision:

```
IF
        The customer has high-risk tolerance
AND
        Meeting objectives requires rapid growth
THEN
        Mutual Fund X is a good choice
```

The Inference Engine has found a **potentially** useful rule, but without more data it cannot determine if this rule **should** be used. To make a further determination, it needs to know if *"The customer has a high risk tolerance".* Determining if this statement is true becomes the new goal of the Inference Engine. The original goal is not forgotten, but it is temporarily superseded by the new goal. The Inference Engine now looks for a rule that can tell it something about risk tolerance. It finds:

IF
The customer's risk ranking is greater than 15

THEN
They have a high-risk tolerance

To use this rule, the Inference Engine needs to know the customers risk tolerance ranking score, which becomes the new top-level goal. This might come from a database, a different program or other rules. The Inference Engine determines where and how to get the needed data. This process of having one goal requiring data, which leads to another goal, can be repeated many times. This "chain" of goals going backwards from the highest level to the lowest level is what gives backward chaining its name.

As data becomes available, lower level goals are met and are dropped off the chain until the Inference Engine is able to determine when the initial top-level goal is met, and the recommendations are presented to the user. A typical one-on-one consultation with an expert takes several paths before reaching a conclusion, but without asking redundant questions. Backward chaining in expert systems emulates this process.

Exsys CORVID also supports another way to run the Inference Engine – **forward chaining**. Forward chaining is data driven, rather than goal driven. Running the Inference Engine in this mode is done when there is a body of data already available and you just want to use the logic in the rules to analyze it.

In this case the rules are tested sequentially to see what conclusions result. Forward chaining is somewhat faster for some problems, and the questions are not as focused. While it is not as good an emulation of a session with a human expert, it is a very effective way to handle data-intensive decisions.

Confidence Factors

Another powerful feature of Exsys CORVID is that the goals can include a "confidence factor" for a particular answer. This enables expert systems to make multiple recommendations with differing degrees of confidence to reach a "best fit" in its conclusion. While in some cases, it is possible to give a specific recommendation with absolute precision; the real world is not often so clear-cut. Often multiple recommendations are simultaneously possible and the system ranks them based on relative merit.

The ability to handle confidence factors in CORVID expert systems provides a much more effective way to build systems that emulate the real world and give the type of recommendations that human experts would. Exsys CORVID provides many different ways to handle and utilize confidence values.

CORVID Variable Building Blocks

CORVID's approach to variables is based on over 20 years of working with customers building expert systems. The object-structured approach makes it much easier to build systems, and overcomes many of the limitations of other systems that required complex syntax or were restrictive in function. The use of properties produces a consistent and flexible way to use variables. This approach is also very easy to extend – as new properties are added, they automatically apply everywhere in the system.

What is a CORVID Variable?

Variables are the building blocks that are used to develop expert systems with CORVID. You can think of them as the elements that would need to be incorporated into a decision-making process. For instance, if a process will use temperature to help make the decision, there will need to be a Variable [TEMPERATURE] defined and used when you build the logic.

Variables are used:

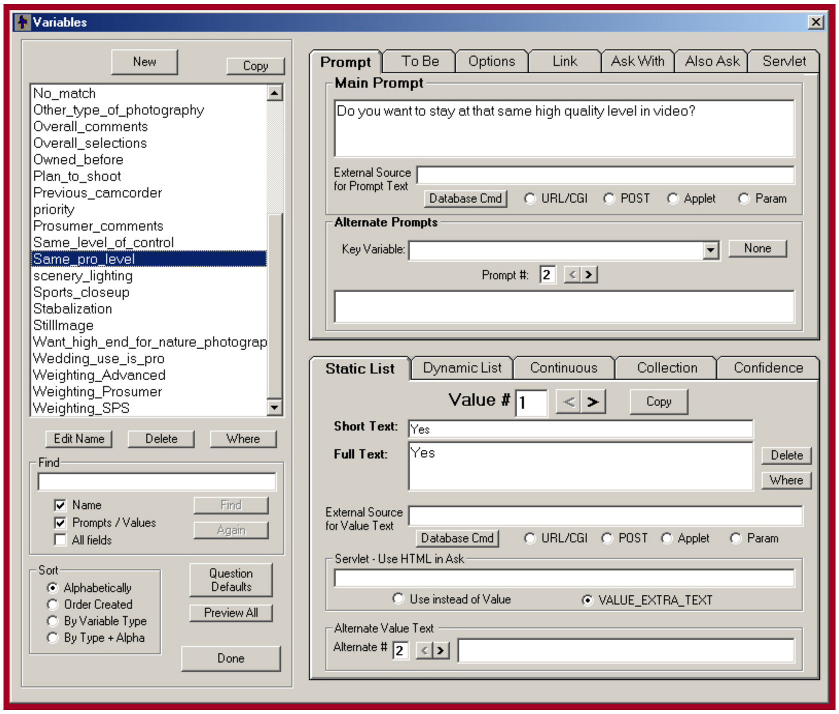
- To define the logic in Logic Blocks and Command Blocks
- To hold data during the execution of the system
- To define the goals of how the system will run

Variable Types

Exsys CORVID provides 7 types of Variables. All of the types of Variables share some characteristics and functions, but each type has special functionality and capability.

CORVID Variable Types	Functions & Capabilities
Static List	Multiple-choice list with the values defined during development of the system. Examples – day of the week, on/off, high/medium/low.
Dynamic List	Multiple-choice list with the values defined dynamically during runtime. The values may come from external sources such as spreadsheets or be set by the logic of the system. Examples: selection of options that change frequently and are not known until runtime.
Numeric	A numeric value that can be used in formulas or test expressions. Possible values are any numeric value within a specified range. Examples: temperature, pressure, stock price, interest rate.
String	A string value that can hold any text string. Examples: name, social security number.
Date	A date value that can be used in comparison (future/past, etc.) tests. Examples: birth dates, option maturity date.
Collection	A list of strings as values. The list is built-up during a run. Various operations allow you to add, remove and test items in the list. Any string, Variable or file can be added to the Collection. Examples: "best" products, configuration, overall comments, or a report.
Confidence	A Variable that can be assigned a confidence value that reflects a degree of certainty. Various formulas can be used to combine the values assigned to an overall confidence for the Variable. Example: likelihood that a product is appropriate for the user.

Variables are added to the system and edited using the Variable Edit Window, which has many options. The left side is to find and select the Variable desired. The top set of tabs are for features which are common to all types of Variables. The lower set of tabs is for features unique to each type of Variable.



No_match
Other_type_of_photography
Overall_comments
Overall_selections
Owned_before
Plan_to_shoot
Previous_camcorder
priority
Prosumer_comments
Same_level_of_control
Same_pro_level
scenery_lighting
Sports_closeup
Stabalization
StillImage
Want_high_end_for_nature_photograp
Wedding_use_is_pro
Weighting_Advanced
Weighting_Prosumer
Weighting_SPS

☑ Name
☑ Prompts / Values
☐ All fields

● Alphabetically
○ Order Created
○ By Variable Type
○ By Type + Alpha

Main Prompt: Do you want to stay at that same high quality level in video?

Prompt #: 2

Value # 1

Short Text: Yes

Full Text: Yes

○ Use instead of Value
● VALUE_EXTRA_TEXT

Alternate # 2

CORVID Window for Adding Variables

Variable Names and Prompts

Each Variable has a **name** and at least one **prompt**.

Name - a shorter way to refer to the Variable and the form used in the Logic Blocks, making the logic easy read and understand.

Prompt - a longer text explaining what the variable means and used when asking the system user for input or in displaying results.

CORVID allows multiple prompts to be specified for a Variable along with a “flag” Variable that selects among them. In this way, it is easy to have a system that can be run in multiple languages just by adding multiple prompts. This can also be used to build a system that can be run by users with different levels of experience – an expert level that asks questions in more technical terms, and a novice level that asks questions phrased in simpler "layman's" terms. This can be especially useful when a company’s professional advisors, as well as their clients run a system.

Variable Properties and Methods

There are many Properties and Methods that can be used with CORVID variables. The use of this object-structured approach to variables makes it much easier to build logic and reports. The properties allow customization, and formatting of data.

A Variable can be assigned a value. The type of variable determines the value that can be assigned –

- Numeric Variables are given numeric values
- List Variables are given one or more values from a list
- Confidence Variables are given a confidence value
- Collection Variables are used to process a list of strings for a report or many other uses

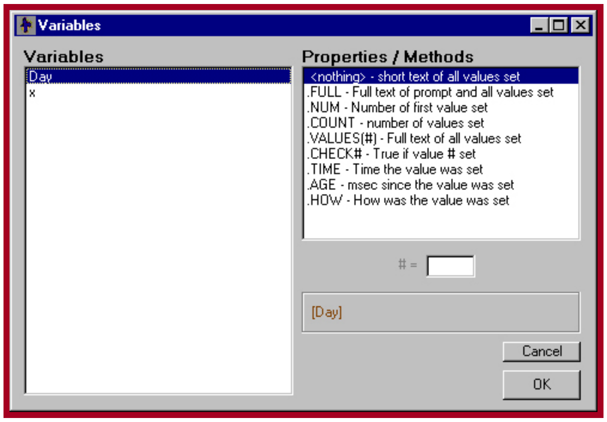
In addition to the value of the variable, each Variable type also has a variety of properties and methods that allow other information about the variable to be obtained or set. This approach provides a very adaptable and extendable way to handle variables in the system logic and reports.

Variables are specified by their name in square brackets. To specify a property, the name is followed by **.property**.

For example, suppose there is a Static List Variable named [Desired_features], and the user can select multiple items from a list.

If the logic of the system depends on how many items the user selected, a formula can use the COUNT property. [Desired_features.COUNT] would return the number of selected items.

If the system needed to show how the value for a variable was set, the .HOW property can be used. [Desired_features.HOW] would return a detailed text explanation of how the value for the variable was set that could be used in the results or a report.



Variables:

Day
x

Properties / Methods:

<nothing> - short text of all values set
.FULL - Full text of prompt and all values set
.NUM - Number of first value set
.COUNT - number of values set
.VALUES(#) - Full text of all values set
.CHECK# - True if value # set
.TIME - Time the value was set
.AGE - msec since the value was set
.HOW - How was the value was set

[Day]

CORVID Property Window for Building Property Expressions

Methods are represented in a similar way, but test or change the value of a Collection variable. For example, a Collection Variable has many methods for adding items to the list - adding entire files, checking if a value is in the list, sorting the list, etc.

To add the contents of a file to a collection variable [RESULTS], just use [RESULTS.ADDFILE] myFile.txt . To add a string “abc” use [RESULT.ADD] abc .

System Goals

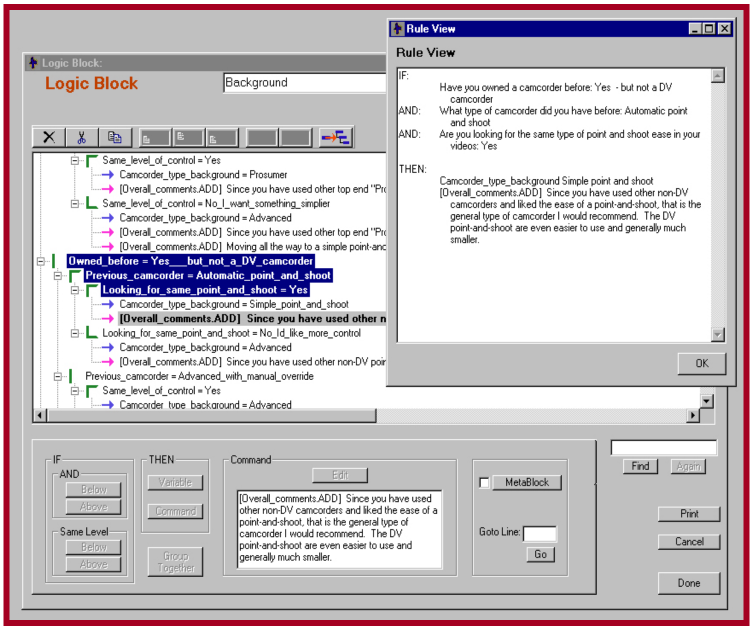
Variables are used in many ways within Exsys CORVID. All Variables can hold data and be used in expressions. Any Variable can also be a system goal. In CORVID, the developer has complete freedom to assign and use the most appropriate variable type as a goal for the system.

If a system requires selecting an item from a multiple-choice list as the goal, simply adding a command that makes a Static List variable a system goal can do it. If the system is designed to build a complex report in a Collection variable, it is just as simple to make a Collection variable the goal. Many systems use Confidence variables as the goals to build systems that select among multiple options by assigning probability to each option. This too is easily done in CORVID.

Building the Logic

Exsys CORVID introduces a unique way to define, organize and structure rules into logically related blocks. The **Logic Block** is made up of one or more tree structured logic diagrams. The logic may be a complex branching tree that systematically covers all possible input cases or a simple diagram that corresponds more to a single rule. The rules in the Logic Block usually all relate to one aspect of the decision, and can be referenced in the system by simply calling that block.

For example, suppose the logic to derive a particular fact calls for 3 multi-branched trees and several other individual rules to cover special cases. A single Logic Block can be built with all the required trees and rules. When the fact needs to be derived, this block can be called. Also, since this block behaves as an object for this derivation, it can easily be used in other places in the system. Blocks are created and edited in the CORVID Logic Block window.



Background

```
      Same_level_of_control = Yes
          Camcorder_type_background = Prosumer
          [Overall_comments.ADD]  Since you have used other top end "Pro
      Same_level_of_control = No_I_want_something_simplier
          Camcorder_type_background = Advanced
          [Overall_comments.ADD]  Since you have used other top end "Pro
          [Overall_comments.ADD]  Moving all the way to a simple point-and
Owned_before = Yes___but_not_a_DV_camcorder
   Previous_camcorder = Automatic_point_and_shoot
      Looking_for_same_point_and_shoot = Yes
          Camcorder_type_background = Simple_point_and_shoot
          [Overall_comments.ADD]  Since you have used other n
      Looking_for_same_point_and_shoot = No_Id_like_more_control
          Camcorder_type_background = Advanced
          [Overall_comments.ADD]  Since you have used other non-DV poir
   Previous_camcorder = Advanced_with_manual_override
      Same_level_of_control = Yes
          Camcorder_type_background = Advanced
```

```
IF:
        Have you owned a camcorder before: Yes  - but not a DV
          camcorder
AND:    What type of camcorder did you have before: Automatic point
          and shoot
AND:    Are you looking for the same type of point and shoot ease in your
          videos: Yes

THEN:
        Camcorder_type_background Simple point and shoot
        [Overall_comments.ADD]  Since you have used other non-DV
          camcorders and liked the ease of a point-and-shoot, that is the
          general type of camcorder I would recommend.  The DV
          point-and-shoot are even easier to use and generally much
          smaller.
```

[Overall_comments.ADD] Since you have used other non-DV camcorders and liked the ease of a point-and-shoot, that is the general type of camcorder I would recommend. The DV point-and-shoot are even easier to use and generally much smaller.

Logic Block and Rule View

The indentation in a block indicates the level of the IF condition in a rule.

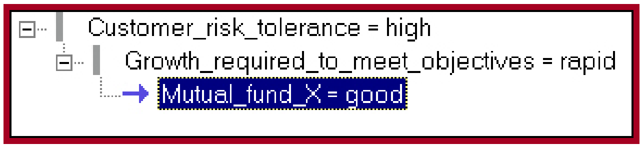
For example, expressing the single rule:

IF
The customer has a high-risk tolerance

AND
Meeting objectives requires rapid growth

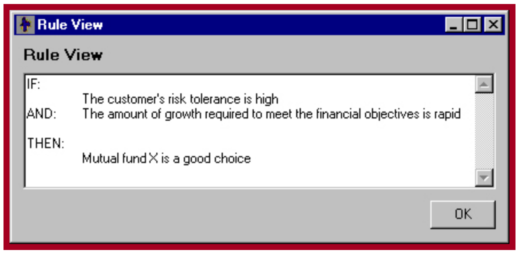
THEN
Mutual Fund X is a good choice

would look like this in a Logic Block.



Customer_risk_tolerance = high
Growth_required_to_meet_objectives = rapid
Mutual_fund_X = good

The horizontal blue arrow indicates the THEN part. The indention of the first two lines and vertical gray bar indicate IF parts. The shorter Variable name is used in the Logic Block, rather than the longer prompt text. When the rule is fully expanded to show the full Rule View (by Shift – Right clicking on the THEN node), it will use the full Variable prompt text and match the original text as shown next.



Rule View

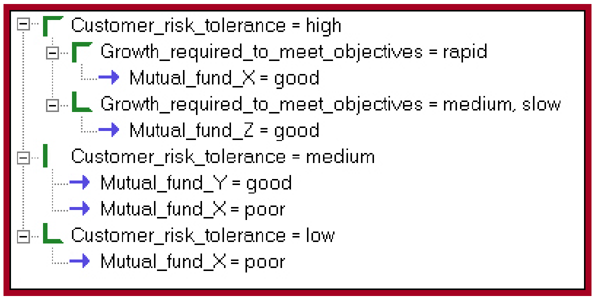
IF:
The customer's risk tolerance is high
AND: The amount of growth required to meet the financial objectives is rapid
THEN:
Mutual fund X is a good choice

Rule View

Tree Structure

In practice, most Logic Blocks are more likely to have some degree of tree structure and cover multiple related cases. The following Logic Block handles more criteria of objectives and goals.

The green angle and vertical lines indicate groups of values for the same Variable, and help organize the block and make sure that all relevant values are considered.



```
Customer_risk_tolerance = high
    Growth_required_to_meet_objectives = rapid
        Mutual_fund_X = good
    Growth_required_to_meet_objectives = medium, slow
        Mutual_fund_Z = good
Customer_risk_tolerance = medium
    Mutual_fund_Y = good
    Mutual_fund_X = poor
Customer_risk_tolerance = low
    Mutual_fund_X = poor
```

This Logic Block is equivalent to the following 4 rules:

IF
The customer has high-risk tolerance
AND
Meeting objectives requires rapid growth
THEN
Mutual Fund X is a good choice

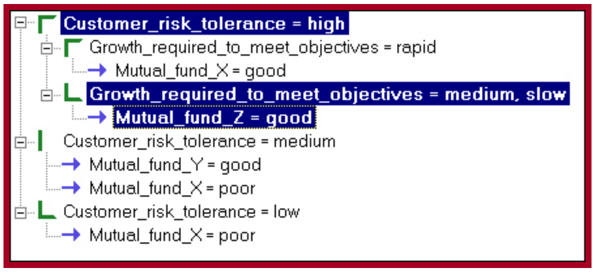
IF
The customer has high-risk tolerance
AND
Meeting objectives requires only medium or slow growth
THEN
Mutual Fund Z is a good choice

IF
The customer has medium risk tolerance
THEN
Mutual Fund Y is a good choice
AND
Mutual Fund X is a poor choice

IF
The customer has low risk tolerance
THEN
Mutual Fund X is a poor choice

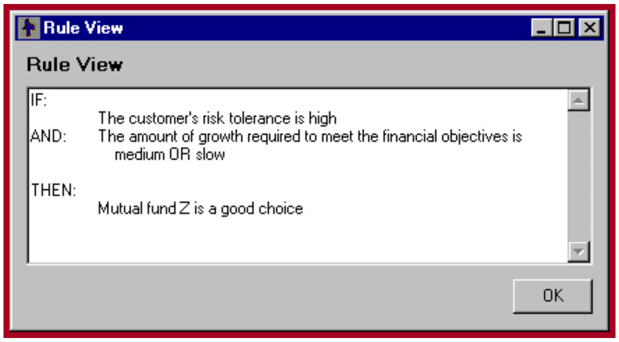
It is also easy to see the structure of a particular rule by right clicking on a THEN condition. This highlights the rule's IF conditions in blue.

For example, for the second rule it would display:



```
Customer_risk_tolerance = high
    Growth_required_to_meet_objectives = rapid
        Mutual_fund_X = good
    Growth_required_to_meet_objectives = medium, slow
        Mutual_fund_Z = good
Customer_risk_tolerance = medium
    Mutual_fund_Y = good
    Mutual_fund_X = poor
Customer_risk_tolerance = low
    Mutual_fund_X = poor
```

A click on a THEN condition brings up a window displaying the full text of the rule as shown below.



```
IF:
          The customer's risk tolerance is high
AND:      The amount of growth required to meet the financial objectives is
             medium OR slow

THEN:
          Mutual fund Z is a good choice
```

Rule View

The unique Logic Block approach of CORVID allows the rules to be used as groups and makes maintenance of complex systems and changes in logic much easier to do.

Exsys CORVID MetaBlocks and Selection Systems

CORVID Logic Blocks also support **MetaBlocks**, a unique capability that provides a way to apply high-level logic to data in a spreadsheet. MetaBlocks provide a way to efficiently build product selection expert systems. The Logic Block contains "generic" logic on how to decide among products based on users requirements. This logic is written in an abstract way that uses product parameters, rather than specific data for an individual product.

For example, a MetaBlock rule might say: *If the customer's budget is less than half the price of the product, the product should not be recommended*:

```
IF
          The customer budget is less than 50% of the product price

THEN
          The product should not be recommended
```

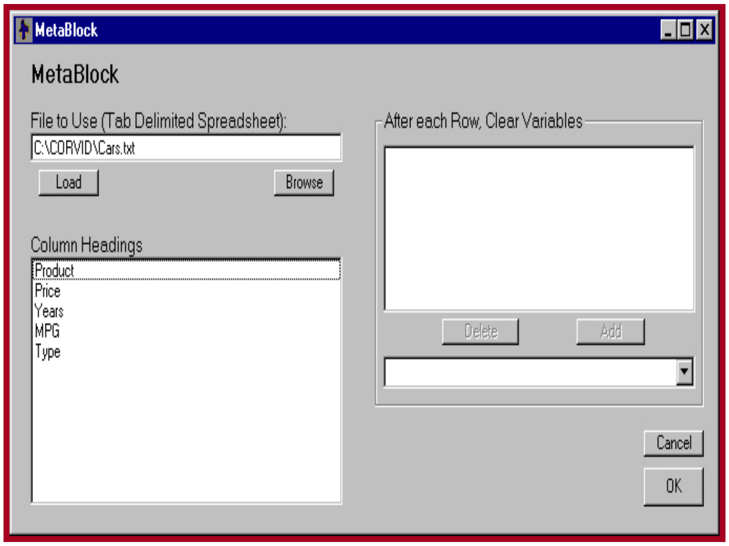
This is a general rule and does not "hard code" any price information on a specific product. In MetaBlock form this would be

```
IF
          [Customer_Budget]   < .5 * {Price}

THEN
          [Product_ranking] = -100
```

The detailed product information is maintained in a spreadsheet. The "Price" parameter in {}, will be replaced by the value from the "Price" column from the spreadsheet for each specific product. The MetaBlock rules can be applied sequentially to data on many products stored in a spreadsheet. As each row (product) in the spreadsheet is analyzed by the MetaBlock, the products can be ranked and sorted to find the best matches to the users requirements.

The logic in a MetaBlock has all of the features of any Logic Block, but can also include the special MetaBlock parameters that reference data from a spreadsheet.



MetaBlock

File to Use (Tab Delimited Spreadsheet): C:\CORVID\Cars.txt

Column Headings

- Product
- Price
- Years
- MPG
- Type

After each Row, Clear Variables

MetaBlock Window

After each row is processed, certain data may be saved or cleared depending on the nature of the system. CORVID has a variety of ways to build reports and recommendations as it analyses the various products.

The overall selection and recommendation of a product can be based on many factors (budget, user requirements, compatibility with other products, etc). A product may not fit every user requirement, but still be the best choice. MetaBlocks find the best fit and also provide a way to explain to the user why this is the best choice.

MetaBlocks also provide an easy way to provide customized notes to the user on each product. If a product is an ideal match on desired functions, but 20% over budget, this could be explained to the user. Very specific notes like this are easy to add and provide a highly customized set of recommendations that add credibility to the system recommendations.

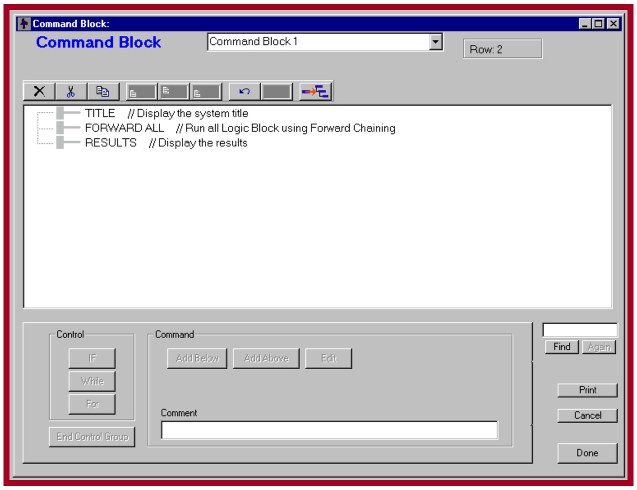
As new products need to be added, or product features change, all that has to be done is updating the spreadsheet. As long as the generic logic for selecting a product does not change, the MetaBlock rules stay the same. This makes it easy to maintain systems even when the products are rapidly changing. A system might even consider inventory, delivery time and other dynamic real-time factors in making a decision. Also, since a URL references the “spreadsheet”, it can be dynamically created. External programs can be called that return data in a tab-delimited form allowing real-time product data to be used for the decision.

Exsys CORVID Command Blocks

The Logic Blocks in a system tell it **how** to make a decision. Command Blocks tell the system what actions to take, and in what order to perform actions.

Command Blocks control the procedural flow of the system including setting what variables are the “goals” of the session, how the system chains, the Logic Blocks to be executed, how results will be displayed and external interfaces.

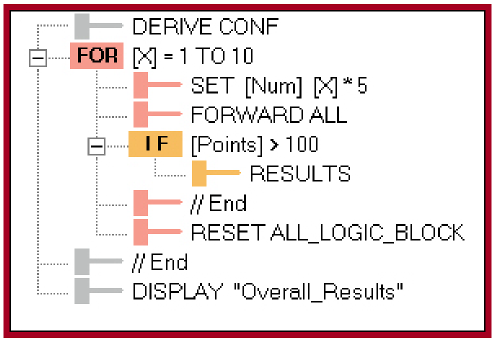
Command Blocks can be a single command that starts backward chaining on all Confidence Variables; up to much more complex systems that use While and For loops, conditional branching, displaying intermediate results, etc.



```
TITLE    // Display the system title
FORWARD ALL    // Run all Logic Block using Forward Chaining
RESULTS    // Display the results
```

Command Block Window

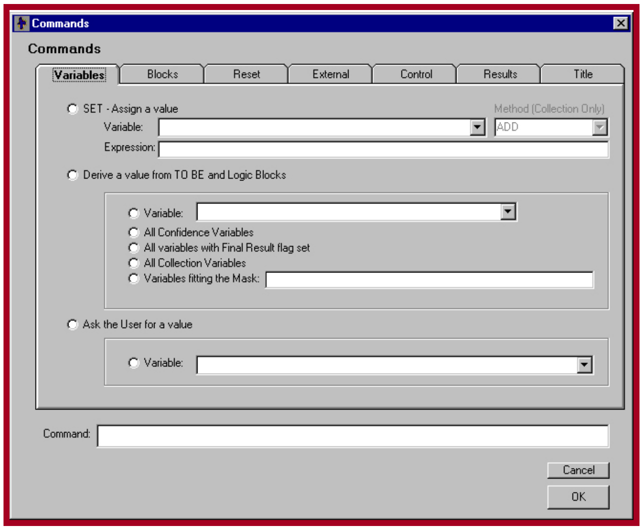
The Command Block provides a graphical development interface to describe the procedural operations, no matter how complex they get. Command Blocks are built and edited in the CORVID Command Block window. This window displays the command structure in a visual interface.



```
DERIVE CONF
FOR [X] = 1 TO 10
    SET [Num] [X] * 5
    FORWARD ALL
    I F [Points] > 100
        RESULTS
    // End
    RESET ALL_LOGIC_BLOCK
// End
DISPLAY "Overall_Results"
```

Conditional branches and loops are color coded and easy to see.

The Command Builder window enables all of the commands to be easily built with a few mouse clicks – no complex syntax to learn, or remember. Using this dialog helps insure that the commands are syntactically correct. Each command also can have an optional note, making the Command Block even easier to understand.


Commands
Commands
Variables
Blocks
Reset
External
Control
Results
Title
○ SET - Assign a value
Method (Collection Only)
Variable:
ADD
Expression:
○ Derive a value from TO BE and Logic Blocks
○ Variable:
○ All Confidence Variables
○ All variables with Final Result flag set
○ All Collection Variables
○ Variables fitting the Mask:
○ Ask the User for a value
○ Variable:
Command:
Cancel
OK

Command Builder Window

Variables Tab - Builds commands that set or drive the value for a Variable, or force the Variable to be asked of the user.

Blocks Tab - Builds commands that run a Logic Block in forward chaining mode or execute other Command Blocks.

Reset Tab - Allows data or blocks to be cleared for reuse.

External Tab - Allows commands to be added that calls other applets, read data from files, write reports and call external programs.

Control Tab - Provides ways to control the flow of execution and include/exclude blocks from the backward chaining.

Results Tab - Provides ways to display the results of a system.

Title Tab - Allows the creation and display of a title at the start of a run.

Controlling The User Interface

The way an expert system interacts with the user, and integrates into a Web site or MIS system is critical to system success. CORVID provides a wide range of ways to customize the look and feel of the system.

The CORVID Runtime programs interact with the users by displaying information screens, asking questions, displaying messages or results, and writing reports. The user interface can be designed using HTML commands and run with the Servlet Runtime, or run from within an applet window with the interface designed using CORVID's internal screen commands.

Questions

Most expert systems ask the user questions in order to get the information needed to process and execute the logic. CORVID provides a wide range of options to control how a question will be asked.

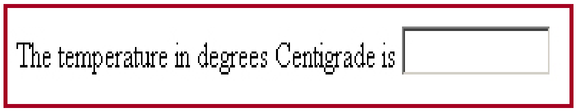
Servlet Templates

In the Exsys CORVID Servlet Runtime, HTML forms are used to ask questions. This makes the full flexibility and power of HTML available for interface design. Question screen are created dynamically based on templates that combine HTML with CORVID commands and replaceable parameters. This allows a "generic" template to be created that defines the page design. Design changes are as simple as editing the template in any standard HTML editor.

For example, to ask a numeric variable the template might include:

```
<!-- CORVID_ASK NUMERIC -->
VARIABLE_PROMPT <input type="text" name="[VARIABLE_NAME]
size="12">
<!-- ASK_END -->
```

The CORVID_ASK NUMERIC command, added to the page as an HTML comment, indicates that this section is to be used to ask numeric variables. The replaceable parameter VARIABLE_PROMPT will be replaced by the actual prompt text for the specific variable. The replaceable parameter VARIABLE_NAME will be replaced by the name of the variable. This template could be used to ask for any numeric variable in a system. Using it to ask for a temperature would look like:



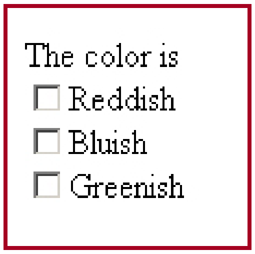
The temperature in degrees Centigrade is

To ask the user to select among a multiple-choice list (a Static List variable), a system might use a template that includes:

```
<!-- CORVID_ASK STATIC_LIST -->
VARIABLE_PROMPT <BR>
<!-- CORVID_REPEAT -->
<input type="checkbox" value="VARIABLE_VALUE_SHORT"
       name="[VARIABLE_NAME]"> VARIABLE_VALUE_TEXT <BR>
<!-- REPEAT_END -->
<!-- ASK_END -->
```

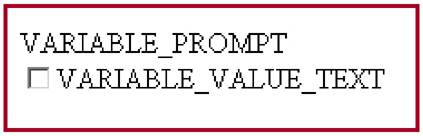
The CORVID_ASK STATIC_LIST command indicates that the section is to be used to ask multiple-choice value questions. The VARIABLE_PROMPT is again replaced by the text of the prompt. The CORVID_REPEAT command indicates that the following lines, up to the REPEAT_END, should be repeated for each value in the variable's value list.

If this template was used to ask for a Static List variable [COLOR] that had 3 values: "Reddish", "Bluish" and "Greenish", it would look like:



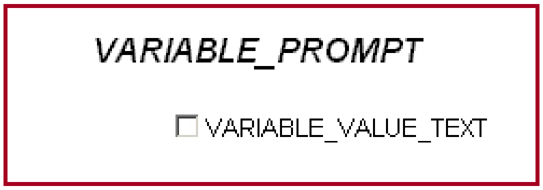
The color is
□ Reddish
□ Bluish
□ Greenish

If the template were opened in an HTML editor, it would appear as:



VARIABLE_PROMPT
□ VARIABLE_VALUE_TEXT

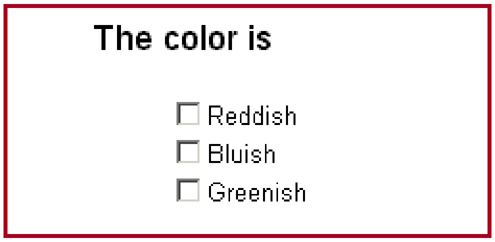
The replaceable parameters are shown as normal HTML text and can be formatted using the editor's commands. The formatting will carry through when the replaceable parameters were replaced by the actual text. If the template was formatted:



VARIABLE_PROMPT

□ VARIABLE_VALUE_TEXT

The question would look like:

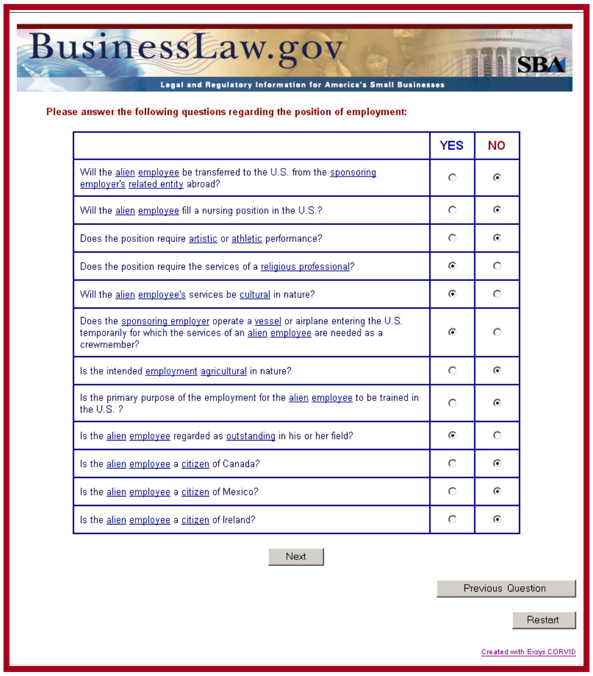


The color is

□ Reddish
□ Bluish
□ Greenish

Building and editing templates is as easy as building the HTML form and adding the CORVID commands as comments. CORVID includes a set of templates that ask questions in different ways. These can be edited to match the design of your site.

Templates allow even complex screens that ask multiple questions to be rapidly created.



BusinessLaw.gov
SBA
Legal and Regulatory Information for America's Small Businesses

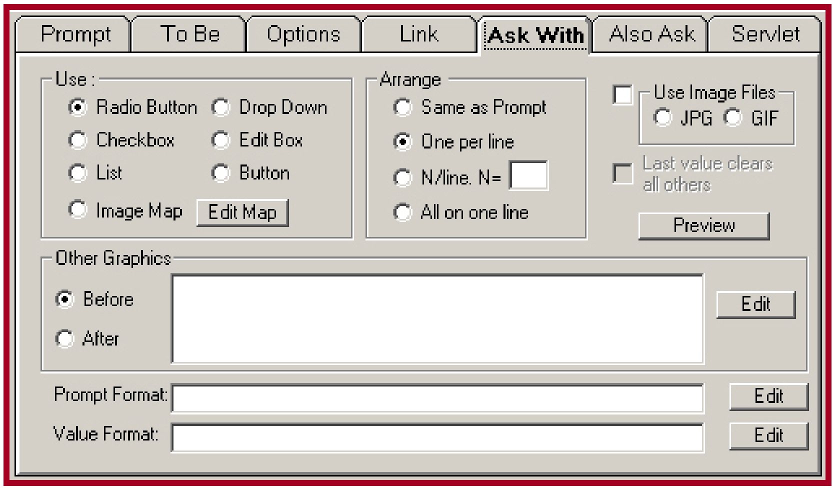
Please answer the following questions regarding the position of employment:

	YES	NO
Will the alien employee be transferred to the U.S. from the sponsoring employer's related entity abroad?	○	●
Will the alien employee fill a nursing position in the U.S.?	○	●
Does the position require artistic or athletic performance?	○	●
Does the position require the services of a religious professional?	●	○
Will the alien employee's services be cultural in nature?	●	○
Does the sponsoring employer operate a vessel or airplane entering the U.S. temporarily for which the services of an alien employee are needed as a crewmember?	●	○
Is the intended employment agricultural in nature?	○	●
Is the primary purpose of the employment for the alien employee to be trained in the U.S. ?	○	●
Is the alien employee regarded as outstanding in his or her field?	●	○
Is the alien employee a citizen of Canada?	○	●
Is the alien employee a citizen of Mexico?	○	●
Is the alien employee a citizen of Ireland?	○	●

Servlet Question Screen that Asks Multiple Questions

Applet Runtime

When using the Applet Runtime, each variable can be asked in a different way by setting parameters from the “Ask With” tab.



Use :
● Radio Button ○ Drop Down
○ Checkbox ○ Edit Box
○ List ○ Button
○ Image Map

Arrange
○ Same as Prompt
● One per line
○ N/line. N=
○ All on one line

□ Use Image Files
○ JPG ○ GIF

□ Last value clears all others

Other Graphics
● Before
○ After

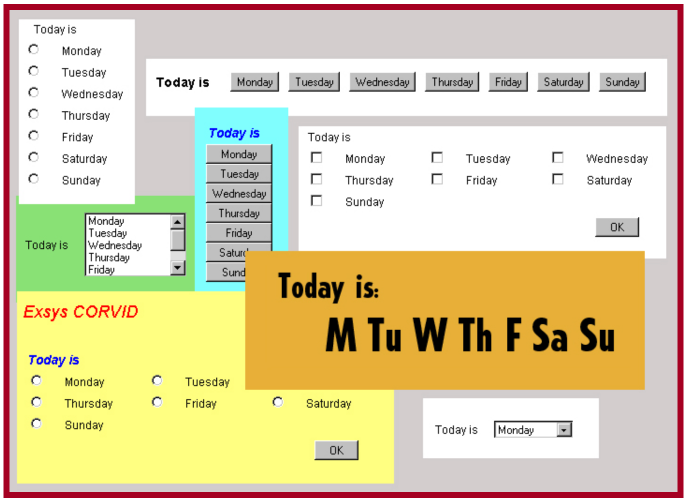
Prompt Format:

Value Format:

“Ask With” Tab for Interface Design

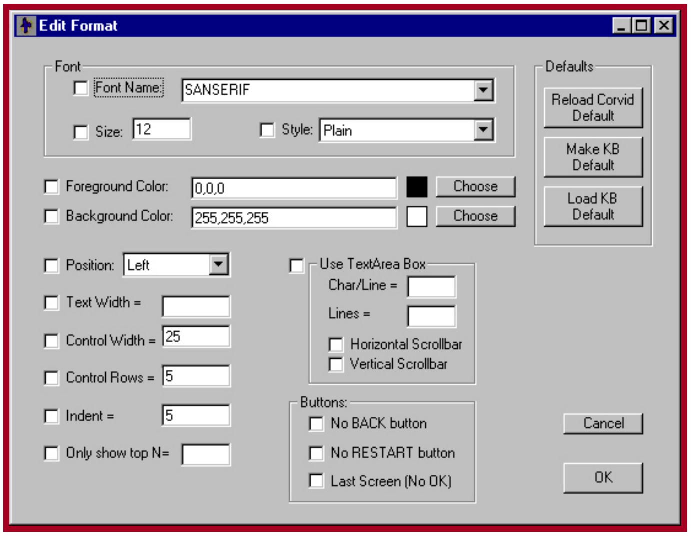
The developer can select the control to use from radio buttons, checkboxes, lists, drop down lists, edit boxes, buttons and image maps. The controls can be arranged in various ways on the screen.

If graphical images are preferred, either a specific image can be displayed for each possible value, or an overall image map can be used. The image map can contain "hot spots" for multiple variables and even use hot spots as "sliders" to set relative values based on where in the hot spot the user clicked.



Various ways to ask the same question

The Format for the text used when the question is asked can be controlled, along with text placement and links. Any text can contain HREF commands setting up HTML links to other pages from text or embedded images.


Edit Format

Font

□ Font Name: SANSERIF

□ Size: 12 □ Style: Plain

□ Foreground Color: 0,0,0

□ Background Color: 255,255,255

□ Position: Left

□ Text Width =

□ Control Width = 25

□ Control Rows = 5

□ Indent = 5

□ Only show top N=

□ Use TextArea Box

Char/Line =

Lines =

□ Horizontal Scrollbar

□ Vertical Scrollbar

Buttons:

□ No BACK button

□ No RESTART button

□ Last Screen (No OK)

Edit Format Window

Links from the text can cause other HTML pages to be displayed in a new browser window, or in a named frame on the same page. Pages can use multiple frames to display various details on a question as it is asked.

Image maps are the most flexible of all ways to ask questions and are supported in both the Servlet and Applet Runtimes. An image (JPG or GIF) can be used. “Hot Spots” are added to the image to mark selection regions. A click in these regions will return a value for a variable when the system is run. The Image Map can have overlapping regions associated with multiple variables and links to other HTML pages. The Image Map editor makes it easy to add image maps to a system.

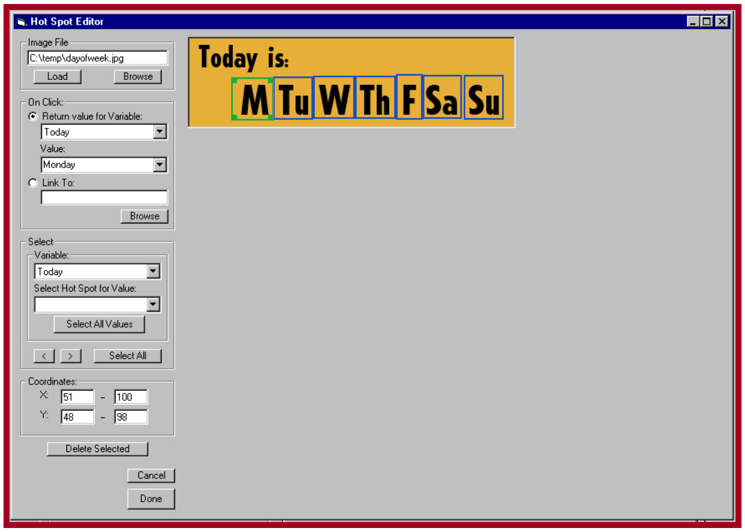


Image File: C:\temp\dayofweek.jpg

On Click:
● Return value for Variable: Today
Value: Monday
○ Link To:

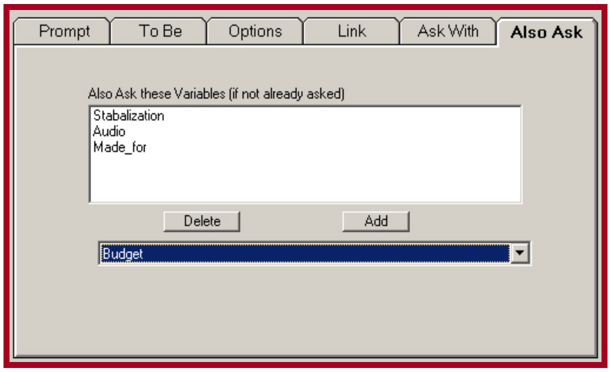
Variable: Today
Select Hot Spot for Value:

Coordinates:
X: 51 – 100
Y: 48 – 98

CORVID Image Map “Hot Spot” Editor Window

Grouping Questions

In many systems, it is desirable to be able to group questions together. In many tools, this is difficult to do, but in CORVID it is easy to do this for each variable using the “Also Ask” window.



Also Ask these Variables (if not already asked)

Stabalization
Audio
Made_for

Budget

Also Ask Window

When a variable is asked, all of the other variables in its “Also Ask” list are included on the same screen. Building complex forms that ask many questions on a single screen has never been easier.

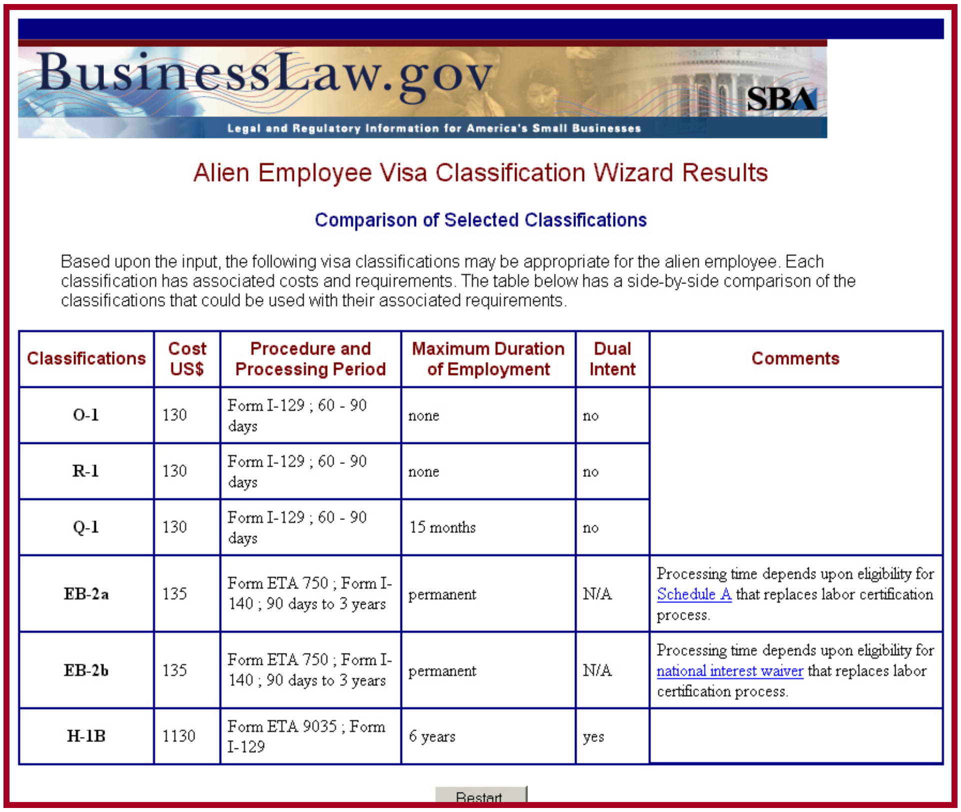
Variables are only asked if they do not already have a value, so nothing is asked twice. Since each variable behaves as an object, it is asked with whatever controls, images and styles are defined for that variable.

The Servlet Runtime uses the appropriate section of the template to ask each question on the same screen. The Servlet Runtime templates even allow a question to be asked in different ways if the user does not provide the needed input the first time.

Displaying Results

There are many ways to display the results of a session. The values of the variables, graphics and text can be formatted in many ways. They use the properties associated with the variables, which enables the output to be exactly as desired. The Servlet Runtime supports special results templates to format output in complex, graphically pleasing ways using HTML.

The Applet Runtime can also build complex reports, using HTML or RTF formats using collection variables that build a report dynamically as the system runs. This approach enables a high degree of control for detailed reports that can be displayed or saved as attractive HTML pages.


BusinessLaw.gov
SBA
Legal and Regulatory Information for America's Small Businesses
Restart

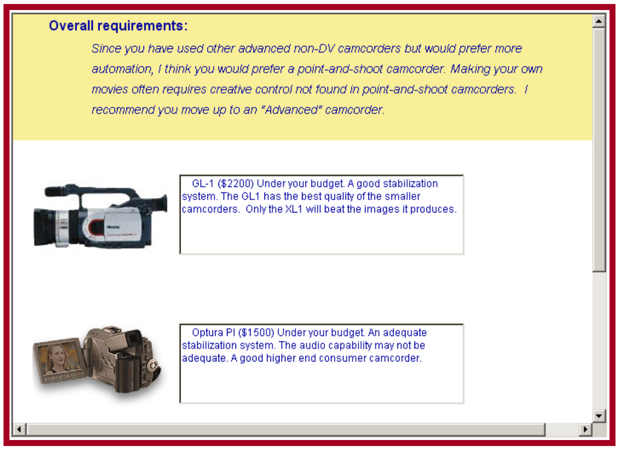
Alien Employee Visa Classification Wizard Results

Comparison of Selected Classifications

Based upon the input, the following visa classifications may be appropriate for the alien employee. Each classification has associated costs and requirements. The table below has a side-by-side comparison of the classifications that could be used with their associated requirements.

Classifications	Cost US$	Procedure and Processing Period	Maximum Duration of Employment	Dual Intent	Comments
O-1	130	Form I-129 ; 60 - 90 days	none	no	
R-1	130	Form I-129 ; 60 - 90 days	none	no	
Q-1	130	Form I-129 ; 60 - 90 days	15 months	no	
EB-2a	135	Form ETA 750 ; Form I-140 ; 90 days to 3 years	permanent	N/A	Processing time depends upon eligibility for Schedule A that replaces labor certification process.
EB-2b	135	Form ETA 750 ; Form I-140 ; 90 days to 3 years	permanent	N/A	Processing time depends upon eligibility for national interest waiver that replaces labor certification process.
H-1B	1130	Form ETA 9035 ; Form I-129	6 years	yes	

Servlet Results Screen Using a Table to Display Recommendations



Overall requirements:

Since you have used other advanced non-DV camcorders but would prefer more automation, I think you would prefer a point-and-shoot camcorder. Making your own movies often requires creative control not found in point-and-shoot camcorders. I recommend you move up to an "Advanced" camcorder.

GL-1 ($2200) Under your budget. A good stabilization system. The GL1 has the best quality of the smaller camcorders. Only the XL1 will beat the images it produces.

Optura PI ($1500) Under your budget. An adequate stabilization system. The audio capability may not be adequate. A good higher end consumer camcorder.

Product Selection Example Displaying Results in an Applet Window

The system results may be linked to other HTML pages for additional details or actions. Images can be used in results to illustrate and explain recommendations. If a system uses MetaBlocks, the spreadsheet file can contain referenced HTML links, and other information that is presented with the results. Product selection systems can recommend the best products for the user, sort them, and allow the user to obtain more information on each one.

Links to Frames

Exsys CORVID provides a variety of ways to link text and images to other HTML pages. This can be done via image maps, links assigned to controls, and HREF commands in strings. Normally, when a link is clicked on, it will open a new browser window to display the associated page. Alternatively, the CORVID applet can run from an HTML page that has multiple frames, and the linked HTML page can be displayed in a named frame on the same page. This allows explanatory information to appear on the same page as the applet itself and enables powerful design options. As questions are asked, CORVID can automatically display information and graphics in the frames.

Databases and Other External Program Access

Databases can be directly accessed from both the Servlet and Applet runtimes using the functions provided with CORVID. The database interface allows the value for a variable to be accessed from a database, and results stored back into a database, and essentially any SQL command to be executed. In addition, other programs can be accessed via calls to CGI, ASP and JSP to process and return the data to CORVID. This opens up a wide range of interface possibilities.

The database access is controlled by a flexible set of commands on the server. Depending on the function required and security considerations, the available commands can be expanded by the system developer. The Java source code for the database interface is provided with CORVID. If your system requires special processing or added functionality, the source code can be edited and compiled to replace the distribution database object.

Open Architecture and API

User defined functions and capabilities can be added to the CORVID Runtimes by adding new Java classes to the Inference Engine. This allows you to add special interfaces or mathematical techniques not built into CORVID. A built-in API allows your code to access any of the data in the system. CORVID can be integrated into the most specialized environments in ways that you can control.

Running in the Development Environment

Both Applet and Servlet interfaces can be tested when a CORVID system is run in the development environment. For applets, the CORVID editor dynamically builds an HTML page. This includes a call to the CORVID Java Runtime applet. The system is run using this HTML page in a browser window. This allows the system to be run in the same environment as on the Web, and it enables the developer to test the system as it is being built.

The Servlet delivery mode can be emulated even when there is no server to test on. The development tool can build the same HTML screens that the servlet will use, allowing a system to be tested with the templates that will be used to field it on the server. If a system is to be run stand-alone as an application, it can also be tested in that mode.

Trace Applet

When developing a system it can be very useful to be able to watch the exact steps the system takes to reach a conclusion. This is particularly helpful when determining why the logic of a system is not behaving as expected.

CORVID allows the addition of the CORVID Trace Applet on the HTML page where all trace information will be displayed. The CORVID
Runtime sends messages to the Trace Applet on the steps it has taken, rules that have fired, how data has been derived, etc. At the end of the run, details on how each Variable received its value are sent to the Trace Applet. In the servlet mode, trace information can be displayed in the questions and result pages while the system is being tested.

This history of the run can be examined and searched to see how and why the system came to the conclusions it did. It is also a way to see certain error messages. This can be especially helpful when there is more than one person involved in developing the system. The trace applet can be run either in development mode or when the system is fielded on the Web.

Extra PARAM to Pass Data

One of the ways to set a value for a Variable is to pass its value in when the applet is called through the applet PARAM option. When the applet is added to an HTML page, various named identifiers can be given values that can be accessed from within the applet. When CORVID builds the call to the applet, it uses this to provide certain information to the applet.

The PARAM option provides a way to pass data using dynamically built HTML pages created through Java Script or an HTML page content manager. These pages can obtain data from other sources before the applet is called.

For example, based on a users login, there may be certain personalization data available, which could be passed directly into the CORVID application and incorporated dynamically without having to re-ask the system user. In some cases, the applet could just appear on the users browser window and reach it’s conclusions without needing to ask any questions.

In the servlet mode, data can be passed with the starting URL, allowing a form or content manager program to build the URL to start the system with initial data.

Stand-alone Systems

Systems can run stand-alone as applications (rather than an Applet or Servlet) to be distributed on a disk or downloaded. In this mode, many of the Java security restrictions are lifted and the program can interact with other programs and files on the client-side, local machine. Systems can also be run in batch mode to process data from other programs. The results can be written to files or passed to other programs.

Even in stand-alone, CORVID’s flexible screen design capabilities may be utilized to ask system questions. The questions will be asked with the same controls and images that would be used from the Applet Runtime. This enables all of your knowledge automation expert system projects to be very flexible in design, and portable in deployment.

Where Exsys CORVID Knowledge Automation Expert Systems are Most Effective

EXSYS CORVID provides ways to capture expert knowledge and decision-making processes in ways that allow computers to emulate the interaction people have with human experts to solve problems. For established, conventional organizations as well as new Web-centric business models, CORVID software makes it very desirable and practical for firms to capture expertise in interactive systems.

Through the use of Exsys CORVID Knowledge Automation Expert Systems, potential customers, support staff, clients and employees are able to obtain problem-solving recommendations, as if they were consulting with a firms' best experts – and they're available 24 hours a day.

Applications are multi-platform adaptable for enterprise-wide integration and run from Web sites, intranets and client-side. They capture expertise and disseminate knowledge to handle mission-critical projects, automate routine tasks, assist in selection processes, analyze data streams, enhance customer/product support, and ensure policy and compliance - any area where consistent, logical decisions are needed.

Best Types of Problems

To best implement expert knowledge in advisory systems, the problem-solving logic should:

- **Be well understood and documented**
- **Be based on logical steps, procedures or business rules**
- **Not involve intuition, guesses, or be based on arbitrary "personal taste" decisions**

EXSYS's approach and techniques of knowledge capture are extremely effective for a broad range of project areas. For over 20 years our customers have successfully implemented thousands of interactive decision-making systems, which are in use worldwide, and they're realizing significant efficiency gains and return on their investment. Some of the most successful categories include:

Successful Knowledge Automation Expert System Categories

Problem-Solving Diagnostics – When experts identify malfunctions or interpret complex data, they quickly look for symptoms indicative of particular problems. The knowledge of how to handle these problems is ideal for conversion into a knowledge automation expert system. The knowledge of the best experts can be made widely available over the Web. By making this knowledge accessible by employees or customers that need it, such systems reduce downtime, increase productivity, and free up experts to handle more complex problems and projects. These systems are also beneficial in capturing the problem-solving expertise of seasoned experts that may be retiring or changing jobs.

Customer/Product Support – These problems are in most ways similar to diagnostic problems, but often go beyond the diagnosis of a malfunction and include following a precise sequence or specific procedures. A knowledge automation expert system is the ideal front-end to a more traditional Help Desk. The system can handle the first (and often higher) levels of interaction with the user. In many cases, the system can solve the problem, without using support staff resources. If the problem is unusual and not part of the system, CORVID can interface to a traditional help desk, opening a case ticket and passing on information about what has already been done to fix the problem. Knowledge automation systems interfaced with traditional help desks bring less experienced staff up to speed quickly without repeated training or interruptions, and provide a more personalized and effective online support environment.

Regulatory Compliance – Though regulations can be very complex, regulations are generally documented in a form very similar to the IF/THEN form of business rules. CORVID regulatory systems insure that all relevant regulations are considered and all policies followed consistently. The logic behind a decision can be automatically documented. These systems provide significant cost savings by helping companies stay within industry compliance, protect employees, and avoid potential fines and possible bad publicity. Many government agencies use Exsys CORVID on their Web sites to provide regulatory information to the public.

WINK-Site Guides – Large Web sites have many pages of information tailored for specific visitors and interests. But actually finding the desired information is increasingly difficult, even in well-structured sites. An alternative to complex linking strategies is a CORVID **W.I.N.K.** (What I Need to Know) system. It provides an “expert consultation” with the visitor to determine what parts of the site will be relevant to them. Then a customized page is dynamically built with just the information they need, and links to the appropriate parts of the site. Unlike some other database techniques that allow limited "customization", with CORVID the full power of the CORVID Inference Engine can be used to determine what to display. The result is a unique and very pleasant experience for visitors.

Decision Support – Business intelligence and advice is becoming a strategic part of many companies' assets and competitive advantage. Many types of decisions and recommendations can be converted into knowledge automation expert systems. They provide business users with the ability to access, analyze, and share information stored in a company’s databases, manufacturing systems and other data sources, in ways not practical with other technologies. Exsys CORVID systems m ake the analytical knowledge of the best experts available to all. Systems can be accessed on-demand, or run in the background to monitor for developing situations. Effectively disseminating and providing access to expert systems greatly increases a firm’s potential in many ways, and is an ideal way to disseminate specialized skills with minimal training.

Smart Questionnaires – One of most common uses of knowledge automation expert systems is to make questionnaires more intelligent. The logic of an expert system leads to asking only pertinent questions, which have been determined to be relevant due to previous answers the user has provided. No irrelevant questions are asked, and no important issues are overlooked. This produces a far superior user “consultation”. Collected data can be sent to other programs, or stored in databases.

Product Selection / Recommendation – Selecting which products best meet a customer's needs and requirements can be a very complex process. But it is one that can be expressed in logical rules relating to customer needs and product specifications. Unlike case-based or "learning" approaches, expert systems handle conflicting requirements by using expert knowledge to weight the factors that are most important for a particular customer or situation. They also always give a recommendation of the best fit, even when all customer desires cannot be met. A Web site is a "virtual salesperson" - it should not say: "we have nothing that matches your needs" - instead it should recommend the best options, just as a human salesperson would.

Configuration – Configuring complex systems with many pieces usually requires an expert with many years of experience. They must determine which pieces are required, but avoid any incompatibilities. This is an ideal problem for the precise and meticulous analysis of an expert system, which can guarantee valid and complete configurations. Modularity allows an Exsys CORVID system to be built in small, manageable modules that automatically combine to provide the high-level logic needed to solve the problem. The system can also create a report explaining the basis for its recommendation. External factors such as inventory and customer deadlines can also be incorporated into the system.

Data Analysis– There are many types of problems, which require a combination of numeric analysis and non-numeric data to be analyzed logically. Knowledge automation expert systems are an ideal way to deal with such problems. Purely numeric analysis can be handled internally, or it can be interfaced to other external programs. The addition of powerful rule-based logic analysis makes it possible to handle non-numeric relationships that are often cumbersome in traditional numerical analysis software. Persons interacting with the systems don't require the extensive knowledge of interpreting the data. The expert system analysis process can even be completely embedded within other systems to appear invisible to end-user.

Inconsistency Detection – Knowledge automation expert systems can monitor customer transactions against policies and procedures to detect inconsistencies that may indicate problems like fraud or other irregularities. The ability of the expert systems to handle complex logic allows such systems to be much "smarter", both in detection and in recognition of illegitimate users that might not trigger other systems.

Manufacturing / Process Control – Knowledge automation expert systems have a long, proven history of use in controlling processes to detect and correct problems before they become serious. From DuPont to Eastman Chemical, many of the major industrial companies rely on expert systems. They run invisibly in the background to analyze various data tags throughout a process and alert operators of potential problems.

Conclusion

Ideally, people would have immediate contact with human experts in every area of specialty that they might need, 24 hours a day. But this can't happen, and many decisions can't wait for or afford access to an expert advisor.

Most organizations have ways to disseminate data. Expert systems developed and deployed with Exsys CORVID directly deliver knowledge - "know-how", advice, and recommendations, rather than just information. This enables people to solve complex decision-making problems without having to learn the underlying logic or needing specialized training. This is the power that automated online expert systems provide – direct delivery of knowledge to the people that need it, when they need it.

Once Exsys CORVID knowledge automation expert systems are built and deployed, the problem -solving skills they contain are available to everyone that needs them. Staff with minimal training can immediately perform at a much higher level. On the Web, anyone can access the systems, and employees that are online can also run the same systems stand-alone. Their easy to use interfaces work well on wireless devices including PDAs, laptops and appliances.

Online knowledge automation expert systems provide traffic-building interaction with your customers, promote customer loyalty and confidence in selecting products, and reduce the load on support staff. Exsys CORVID knowledge automation expert systems provide a very effective and efficient way to provide people - prospects, customers, employees and even advisors themselves, with a way to have 'round-the-clock access to top-level expert decision-making knowledge and consistent advice.

Specific automated dissemination of knowledge is the one of the biggest benefits of knowledge automation expert systems. Capture knowledge, deliver answers!

EXSYS Services

Training

EXSYS Inc. offers training in the use of EXSYS software either on-site at your facility or at the EXSYS Training Center in Albuquerque, NM. Both beginning and advanced training are available. Whether for individuals, project groups, or enterprise-wide standardization, EXSYS training is an ideal way to quickly become familiar with Exsys CORVID features, system authoring and integration.

Consulting

EXSYS Inc. offers a broad range of consulting services to quickly help move your expert system projects to success. Our company has over 20 years of technical expertise that can provide prototyping, project support, knowledge engineering, customization, system conversion and integration.

We can also draw from our technology partners to fulfill your expert system programs from concept through ongoing maintenance. We handle a broad range of problems to ensure the best performance and return on your expert system investment.

Contact an EXSYS representative today to discuss your ideas, find out about our many customer case studies, plan application strategies, and discover the best approaches for efficient and effective expert system projects.

EXSYS®
Capture Knowledge
Deliver Answers™

Headquarter Office:

EXSYS, Inc.
2155 Louisiana Blvd. NE
Suite 3100
Albuquerque, NM 87110
U.S.A.

Tel: +1.505.888.9494
Fax: +1.505.888.9509

info@exsys.com
www.exsys.com