

An Intelligent Information System for Forest Management: NED/FVS Integration

J. Wang, W. D. Potter, D. Nute, F. Maier
Artificial Intelligence Center,
111 Boyd Graduate Studies Research Center,
University of Georgia, Athens, GA, 30605
jwang@ai.uga.edu

and

H. M. Rauscher, M. J. Twery, S. Thomasma, P. Knopp
USDA Forest Service, Asheville, NC and Burlington, VT

Abstract: An Intelligent Information System (IIS) is viewed as composed of a unified knowledge base, database, and model base. This allows an IIS to provide responses to user queries regardless of whether the query process involves a data retrieval, an inference, a computational method, a problem solving module, or some combination of these. NED-2 is a full-featured intelligent information system for the sustainable management of forest lands. It is designed to help managers plan for wildlife, ecology, water, and landscape objectives as well as timber production. FVS is one of the integrated decision support model components in NED-2. We provide the FVS simulation agent and “wrapper” that permit communication between FVS and NED-2. We are developing a meta-knowledge base. Simulation agents can use it to set up and execute external simulation models. This paper will briefly describe the NED decision process, the NED-2 architecture, and discuss the design issues explaining the integration of NED-2 and FVS.

Keywords: intelligent information system, blackboard architecture, agents

1. Introduction

Owners and managers of US forest lands are being challenged as never before to produce an increasingly complex set of benefits as a variety of costs increase. Getting the most money from your forest land through timber harvesting is by itself a very challenging goal. But a growing number of landowners want much more than money from their forest lands. They may want to create or maintain certain desirable ecological conditions such as a grove of “old-growth forest” or a scenic, park-like environment. They may want to restore portions of their property to more natural conditions in order to enhance the habitat for many creatures well into the future. Knowing that one is being a good steward of the land may also be part of the rewards of forestland ownership. And many landowners have said for years that they are more concerned with enhancing wildlife habitat, increasing bird population diversity, or protecting rare and endangered species than they are with high returns from timber harvesting. Land owners want to manage their forest lands for more and more goods, services, and

environmental conditions than ever before. However, providing for these increasingly complex benefits often increases management costs and the cost of ownership.

Researchers at the USDA Forest Service, Northeastern and Southern Research Stations, in cooperation with many collaborators, have been developing a computer software system called the NED Decision Support System. This system is designed to help forestry consultants and their private landowner clients develop goals, assess current and potential conditions, provide a way to study and select from different scenarios, and produce management plans for their forest properties.

In this paper, we will briefly describe the NED decision process, the NED-2 architecture, and how we have used the theory of Intelligent Information Systems (IIS) to link FVS into the NED-2 DSS.

2. The NED Decision Support Process

The purpose of modern forest management is to achieve diverse goals selected by the landowner. This perspective is critical and central to the NED Decision Process. It cannot be overemphasized that without goals, reasoned management cannot be practiced.

Surprisingly, identifying and choosing a good set of goals is the most difficult part of the entire decision process. A difficult part about choosing suitable goals is that you have to be able to tell whether you have achieved them or not. For example, one of the goals for the Deer Hill case study in South Carolina was to focus on producing *Wild Turkey Habitat*. Well, you can't just walk into the forest, pull out your *Wild Turkey Habitat* measurement gauge, swing it around and get a reading on it. There is, of course, no such gauge and that's because *Wild Turkey Habitat* is an abstract concept that unifies many factors about the birds and their needs. So how do we measure it? Well, first we have to further define what we mean by *Wild Turkey Habitat*. We talked with turkey management experts and decided that we needed three things for good turkey populations: (1) a favorable landscape pattern, (2) forest land with all sizes of trees represented, and (3) turkey food. That list certainly helps, but we are still not able to go out and measure any one of these three things. So what do we do? We define what we mean by each of the three sub-divisions. A favorable landscape pattern for turkey is defined as: (1) park-like, open forests cover at least 80% of the area, (2) scattered small fields make up more than 10% of the area, (3) park-like large hardwood forests are present near small fields, and (4) park-like large pine forests cover at least 10% of the area. So now, finally, we have something we can measure in our forests. By going through this process, we create a list of goals in sequence with immeasurable but valuable top level goals at one end and measurable conditions that define the top level goals at the other end of the hierarchy.

Having defined our goals, we next need to learn about our property. Many landowners do not possess an inventory of the trees on their land and have only a general idea of the mix of vegetation, soil, and topographic features. Obtaining an inventory is an expensive proposition yet a cost that must be paid prior to the beginning of serious efforts to establish a cost effective management program. Knowing this, the NED Team developed a relatively inexpensive and yet quite complete inventory procedure that provides good estimates of the large trees, the

smaller trees, shrubs and herbs, and allows for a rapid assessment of wildlife habitat conditions using measures such as the presence or absence of dead standing and dead fallen trees. We look for perches for birds, whether moss, ferns, or grasses cover the ground, whether there is permanent water available for all creatures and especially salamanders and turtles, and so on. Although we look for many things, we have designed a process that takes approximately ½ hour per forest stand.

Given the set of goals and an understanding of the current forest conditions, we can turn our attention to figuring out what we might do to our land (if anything) so that it better achieves the goals. We want to create a small number of very different strategies for managing our land while satisfying the goals. These strategies are called management scenarios. Each is a different road to get to the same place. In sustainable forest management there is rarely a single, best road to follow to achieve a given set of goals. What we can realistically do is design several different ways (roads) to get to our goals and then compare them. At each cycle through this process, we learn more about our own values and goals, about our land, and about the things we can do with it. For Deer Hill, we designed three management scenarios (Figure 1):

- (1) A Timber Only scenario: maximum profit from timber operations consistent only with best management practices for sustainable timber management and the CRP requirements. Wildlife is not specifically addressed and no revenue from wildlife operations is expected. All open areas will be planted to loblolly pine, the pine-hardwood stands will be commercially chipped and converted to loblolly pine plantations as soon as feasible, loblolly pine plantations will have two thinnings (age 10-12 and 20-25) and a final harvest at age 30-40, and the plantation size class distribution will be spread out to get a more even flow of income.
- (2) A Timber and Extra Products scenario: maximum profit from non-timber, non-extractive human use of the land. Leave 400 acres of existing pine plantations alone and continue to manage for maximum timber. Take 100 acres of pine plantation and manage for big pine over long rotations. The pine-hardwood stands will be turned into park savannahs with large hardwood trees spread out over a 30 by 30 foot grid. Islands of regenerating hardwoods will be created in these open forests. Wildlife food plots will be established in all open fields and the wildlife row-crop planting (corn) will continue. Many different camping, nature education, hunting, and other sporting activities will be developed as income producing activities.
- (3) A Timber and Hunting scenario: maximum profit from timber and hunting operations. Leave all pine plantations as they are and manage for maximum production. Rent hunting rights to highest bidder. Thin pine-hardwood and hardwood stands to promote acorn production. Keep open fields open and food plots productive.

Having developed the three management scenarios for Deer Hill, we had to pretend to carry each of them out over our 40-year time frame and then compare them to each other. We did this by using a forest growth forecasting software package called the Forest Vegetation Simulator (FVS). FVS was created and is maintained by the USDA Forest Service forest management service center in Ft. Collins, CO. This system covers all forest types in the entire United States. It is, however, fairly complicated to use and requires at least one week training

before users feel comfortable with the software. NED integrates FVS but only for regions east of the Mississippi. FVS is available on the Internet for downloading, free of charge, at www.fs.fed.us/fmsc.

We are almost at the end of the NED decision process now. To recap: We set our goals; learned about the current condition of our property; figured out some alternative ways we could manage our land; and projected those alternatives over a 40 year time horizon to get an estimate of how each forest is likely to look 40 years in the future. Now we can go back to our goals, find our measurable conditions, and evaluate them against each of our simulated future forests (Figure 2). For example, in the *Wild Turkey Habitat* goal, we needed park-like hardwood forests with large trees near small fields. By evaluating the three simulated scenarios, we can determine that only the **Timber and Extra Products** scenario (#2) will provide us with that condition. Most producing oak forests, on the other hand, are found in both the **Timber and Extra Products** scenario (#2) and the **Timber and Hunting** scenario (#3). Comparisons were continued for each of the measured conditions. It is then a relatively simple matter to rate each scenario against each measurement condition and then determine which scenario does best in satisfying the top-level goals.

We usually learn a lot from this process. We may find that a goal that we selected at the beginning turns out to be unrealistic. We learn this because no matter what we do in any scenario, we simply cannot achieve this particular goal given the resources we are willing to spend and the time we are willing to commit. We then may wish to change our goals or maybe see if we can achieve them in 60 years instead of 40 and thus leave a legacy for our grandchildren. We may also discover through discussions some other ways to manage our property, thus creating another alternative scenario. Such changes are OK, because now it's inexpensive to play "what-if" games with the NED software; to look at different futures and different goals until we are comfortable with our "final decision". This final decision is, of course to be used, but it is only tentative. It is likely that next year or the year after, we as well as our world will have changed enough that our "final decision" may be outdated. But because we have done our homework, know our land, and have increased our understanding, we can go through the NED process again pretty quickly and efficiently.

We have described the NED decision process. This process is currently being field tested in case studies with real users. The NED-2 decision support system is being developed to automate the NED decision process and make it as efficient and easy to implement as possible. Next, we will provide a brief overview of Intelligent Information Systems theory, a description of the NED-2 architecture, and then discuss the implementation issues about how we integrated FVS into the NED Intelligent Information System.

3. Intelligent Information Systems and the NED-2 Architecture

We view an Intelligent Information System (IIS) as composed of a unified knowledge base, database, and model base. This allows an IIS to provide responses to user queries regardless of whether the query process involves a data retrieval, an inference, a computational method, a problem solving module, or some combination of these (Potter and others 2000). The integration of information sources, whether local or remote, should be transparent to the

user. That means after getting user queries the system itself decides which source(s) need to be accessed.

In order to add new functionality to a software system, one approach is to integrate additional autonomous, heterogeneous information sources. Since each source is designed to perform a particular task using its own data model, we might need to deal with these issues. First, the data format of the information source may need to be translated in order to communicate with the other sources. Second, the information source that is developed in a different programming environment should be invoked by appropriate control codes. We might also need to consider how to perform transparent processing so that the users would just focus on their queries without having to consider where the results could be gotten. There are many existing approaches that provide intelligent integration of information, such as a federated database approach (Sheth and Larson 1990), a hierarchical mediator approach (Papakonstatniou and others 1996), a description-logic-based approach (Levy and others 1996) or an ontology-based approach (Cheung and Cheng 1996). Instead of designing and building NED-2 based on a monolithic and pre-defined solution, we approach the development by creating a loosely organized system that consists of smaller components. Each smaller component performs its own functions within the larger problem-solving framework. Essentially, every component is an external heterogeneous information source composed of an intelligent agent, its associated knowledge base, and whatever source it accesses. By consulting the knowledge base, an external model management agent knows when and how to perform its tasks. Therefore, the use of intelligent agents and knowledge bases makes integration of heterogeneous information sources easier and more flexible.

NED-2 is a blackboard system with semi-autonomous intelligent agents. Its blackboard integrates a Microsoft Access database and a set of Prolog clauses. Inventory data and other information are stored in the database. NED-2 includes a user interface, databases, simulators, knowledge bases, hypertext documents, geographical information systems and visualization tools. Simulators and other external modules are integrated into NED-2 via their intelligent agents (Figure 3).

In an agent-based blackboard system, each agent makes a contribution to the problem-solving process. Agents communicate with each other through a blackboard. Tasks that need to be done are posted to the blackboard. Agents also post most of the intermediate results of their activities to the blackboard. Agents watch the blackboard continually. The information on the blackboard will prompt an agent to do some work. If an agent performs some task listed on the blackboard, it will erase that task from the task list. An agent may place a report that the task has been performed on the blackboard after it completes the task. If an agent begins a task, then discovers that something needs to be done that is beyond its capability, it can put the new task on the blackboard and wait until another agent performs it before completing its original task.

In NED-2, Prolog, a high-level logic programming language, provides the primary implementation platform for agents, knowledge bases, and inference engines. The user interface is implemented in Microsoft Visual C++. The databases are implemented in Microsoft Access. To perform its tasks, an agent may need to retrieve and update the core data on the blackboard. The Prodata LPA Prolog interface (http://www.lpa.co.uk/ind_pro.htm) is

used to implement the database access. The Prodata interface provides a tight coupling between LPA Prolog for Windows and all Database Management Systems (DBMSs) that support a sufficient level of Open Database Connectivity (ODBC) compliance to be used with Microsoft ODBC 2. Prodata allows database tables to be accessed from Prolog as though they existed within Prolog's environment as unit ground clauses (facts). All routine database functions such as creating tables, updating/retrieving records, and/or whatever may be achieved via normal Structured Query Language (SQL) commands. When an agent needs to get something from the blackboard or put something onto the blackboard, it calls a particular predicate designed to retrieve or update information. If information is already present as facts on the blackboard, it is easy to be accessed or changed. However, if information is stored in the core database, a Prodata SQL query will be constructed to select, update, add, or delete the data values in the database.

4. The FVS Agents, wrappers and meta-knowledge bases

The first growth and yield models integrated with NED-2 are the Northeastern and Southern variants of FVS (Teck and others 1997). Simulation agents are designed to set up and call these growth and yield models. The NED user develops management plans using a drag-and-drop scheduling screen (Figure 4). Treatments are selected and dropped onto individual stands. Management plans are stored in the core database. A set of default treatment parameters and user specified treatment parameters are stored in the core database also. A set of rules is included in the simulation knowledge base. According to these rules, a simulation agent for an external model will determine all the conditions that must be satisfied for the model to function. For example, FVS needs stand information such as stand year of origin, site index species, site index, and basal area. If a condition is not satisfied, the agent will go to the blackboard to see if the message it needs is there. If not, the agent will post a request on the blackboard. Eventually the required message will be posted. Once all conditions are satisfied, the agent will run the external model. Currently the FVS agent needs to invoke the FVS wrapper in order to access FVS. The FVS wrapper creates the FVS key file and FVS tree data file, runs FVS, and then converts the FVS simulation results and inserts them back into the NED-2 database (Figure 5).

4.1 The FVS Agents

There are two agents related to FVS, the baseline agent, and the FVS simulation agent. The baseline agent generates data of the baseline year if necessary (when inventory data are collected in a previous year from the specified baseline year). The baseline agent gets the baseline year value from the blackboard, and then checks every stand in the specific management unit under consideration. If the last inventory year for a stand is the baseline year, the baseline agent does not need to take any action. However, if the inventory year is different from the baseline year, the baseline agent will set up a request for the FVS wrapper to run FVS in order to generate the data for the baseline year.

The primary functions of the FVS simulation agent are to retrieve the user's treatment plan and manage the FVS run via the wrapper. A treatment plan includes which stands will be simulated, how long the simulation will run, how to implement the treatment, and when to treat

the stand. The FVS simulation agent gets the treatment parameters from the blackboard, then requests the FVS wrapper to simulate the plan.

In addition to these primary functions, there are other functions such as determining which variant to use for FVS and in some cases which region within a variant. If this information is not available on the blackboard, the FVS agent will put a request on the blackboard and the variant agent will respond to this request. Currently, the variant agent is responsible for providing the proper FVS variant information via a graphical interface. It is designed to allow the user to select an appropriate FVS variant (and region). We plan to incorporate a knowledge base into the variant agent that would suggest a variant to the user based on known conditions (such as knowing the state where the stand is located, that is, if the stand is in Vermont, it is likely that the northeastern variant would be utilized). If the user chooses a variant that is not consistent with the user's inventory data, the agent will issue a warning window to the user to allow an alternate variant to be selected or a warning override. Once the variant is decided, the agent will post appropriate messages on the blackboard.

4.2 The FVS wrapper

In intelligent information systems, a wrapper is an important component. Usually, information sources are heterogeneous (and sometimes distributed). They could (1) be multimedia (video, sound, images, and text), (2) be stored in diverse formats (databases, flat files, or Internet sites), (3) have different data meanings across sources (e.g., grade point average might be based on a 4.0 system in one database and a 10.0 system in another), (4) differ in temporal and spatial dimension, and/or (5) be application programs. By building wrappers "around" the heterogeneous information sources, we can communicate with them easily. The function of a wrapper should be to accept the caller's queries and data, convert them into the target-source format, and then pass them on to the target-source for execution. After execution, the results are captured by the wrapper that will transfer them into the format of the caller.

NED-2 is written in LPA Prolog and Microsoft Visual C++. Agents, the callers, are written in Prolog. FVS is written in FORTRAN. We provide an FVS wrapper that permits communication between FVS and agents, and is also written in Prolog. Because the input and output of FVS consists of only plain text files, and NED stores all the data in a Microsoft Access database, the FVS wrapper needs to transfer the data between these two formats.

The FVS wrapper includes three main modules: the MDB to FVS module, the FVS to MDB module, and the NED calculation module. It (1) accesses related stand data in the NED-2 database to create FVS keyword and stand files, (2) runs FVS, (3) converts the FVS simulation results and inserts the results back into the NED-2 database, and (4) runs the NED calculation module to calculate simulation results at the stand level.

(1) MDB to FVS Module

This module takes the stand data, plot data, and tree data from the blackboard and creates a keyword file (*.key) and a set of FVS tree data files (*.fvs).

One of the FVS input files is the keyword file. In NED-2, we completely by-pass the SUPPOSE interface that allows direct FVS users to specify treatments and treatment parameters when running FVS. The values that are included in the keyword file can be retrieved from the NED-2 database and treatment plan.

For keyword related thinning, the parameters will be decided by the treatment plan. The MDB to FVS module takes the plan, and then retrieves the corresponding thinning parameter from the treatment parameter database. For example, if the treatment plan is “light thinning from below using basal area in FVS in 2010”, and the default values for light-thinning are residual_ba(120), min_dbh(1), and max_dbh(7) in the treatment parameter database, the keyword thinBBA will look like “ThinBBA 2010 120 1 1 7 0 999”.

(2) FVS to MDB Module

After running FVS, we have results of the simulation. They are saved in a tree list file (*.trl). This module takes the tree list file as input and inserts appropriate data back to the NED-2 database. The FVS to MDB module selects each line in the tree list file. It checks if the data record for this tree is duplicated. If it is, the record is ignored and processing continues with the next tree record. If it is not, the module gets the values of tree ID, tree species code, tree diameter at breast height, and tree stems per acre. These items are converted to the NED-2 database format and sent back to the NED-2 database.

(3) NED-2 Calculation Module

The function of this module is to run the NEDCalc.dll. In NED-2, trees are identified by the stand, cluster, and plot that they belong to. Tree data are stored in the overstory and understory tables according to d.b.h. To facilitate the retrieval from the NED-2 database, the FVS wrapper runs the NED calculation module before calling the MDB to FVS module. The NED-2 calculation module then runs several routines that will create a “pseudo_stand” combining all trees in a cluster into one “pseudo_plot”. After running the FVS to MDB module, the FVS wrapper calls the NED-2 calculation module again. This time the NEDcalc.dll shuffles the simulation results into the overstory and understory tables, and then computes the other tree data at both the tree level and stand level.

4.3 Meta-knowledge bases

To make the simulation agents more intelligent, we are creating a meta-knowledge base for simulation agents. Meta-knowledge is knowledge about knowledge. Different simulators require different formats for input data and generate output data differently. Wrappers can provide data translation. But on the one hand, for each specific simulator, we have to build a corresponding wrapper in order to set up and run the simulator. On the other hand, these simulator wrappers include some common procedural knowledge. Therefore, it is desirable to develop a meta-knowledge base. Simulation agents will use this meta-knowledge base to know when and how to use a knowledge source. For example, by consulting the meta-knowledge base, the simulation agents will know how to convert NED-2 data into the format FVS can accept, how to set up control codes for FVS, and how to convert output from the FVS format back into the NED-2 format. So all procedural knowledge for running simulators will be provided by the simulation agent and specific knowledge needed to run individual simulators will be stored in the meta-knowledge bases.

5. Conclusion

We have described the FVS integration process for NED-2. Our blackboard architecture based on intelligent agents makes integration of information sources easy, fast, and more powerful. Our future work is to build more intelligent simulation agents. Other simulation models, such as Silvah (Marquis and Ernst 1992), will be integrated in NED-2. Simulation agents will consult meta-knowledge to know when and how to access related sources (FVS, Silvah, or other integrated simulators). NED-2 will be able to respond to high level queries, like “Show me how Stand 10 will look in 25 years if I remove all hardwoods under six inches in diameter today.” The simulation agent will reformat the original user query and run an appropriate growth and yield model according to the meta-knowledge base. This allows seamless data base, knowledge base, and model base interaction that is transparent to the user. The point is to increase the intelligence of the software in order to reduce the complexity that the user has to overcome. For further information on NED, please visit NED website at <http://www.fs.fed.us/ne/burlington/ned>.

References

- Cheung, W.; Cheng, H. 1996. The Model-Assisted Global query system for multiple databases in distributed enterprises. *ACM Transactions on Information systems*, 14(4): 421-470.
- Levy, A.; Rajaraman, A.; Ordille, J. 1996. Querying heterogeneous information sources using source descriptions. *Proceedings of the 22nd VLDB Conference*, pp.251-262. Bombay, India.
- Papakonstatniou, Y.; Garcia-Molina, H.; Ullman, J. 1996. MedMaker: A mediation system based on declarative specifications. *IEEE 12th Int. Conference on Data Engineering*, pp. 132-141. New Orleans.
- Potter, W. D.; Deng, X.; Somasekar, S.; Liu, S.; Rauscher, H. M.; Thomasma, S. 2000. Forest Ecosystem Management via the NED Intelligent Information System. *Proceedings of the 13th Int. Conference on Industrial & Engineering Applications of Artificial Intelligence and Expert Systems, IEA/AIE'2000*, pp. 629-638. New Orleans.
- Marquis, D. A., and Ernst, R. L. 1992. User's guide to SILVAH: stand analysis, prescription, and management simulator program for hardwood stands of the Alleghenies. Gen. Tech. Rep. NE-162. Radnor, PA: U.S. Department of Agriculture, Forest Service, Northeastern Forest Experiment Station. 124 p.
- Sheth, A.; Larson, J. 1990. Federated database systems for managing distributed, heterogeneous and autonomous database. *ACM Computing Surveys*, 22, 3, pp. 183-236.
- Teck, R.; Moer, M.; Eav, B.1997. The forest vegetation simulator: a decision-support tool for integrating resources science. <http://www.fs.fed.us/ftp/root/pub/fmsc/fvsdesc.htm>

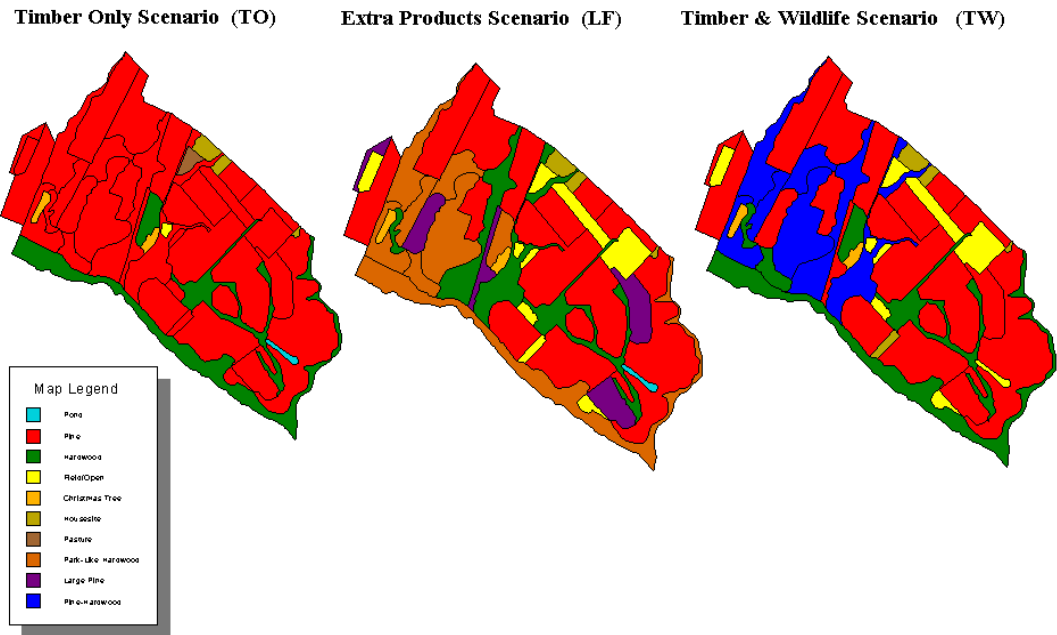


Figure 1: Computer models allow the forest manager to predict the results of different management options

PRIMARY GOALS	SUBGOALS	DESIRED FUTURE CONDITIONS	ALT=TD	ALT=LF	ALT=TW	
1.0 Maximum Net Worth		1.1 Annual Income > \$35,000	75%	80%	90%	
		1.2 Maximum \$\$ Income	\$816,000	\$936,000	\$932,000	
		1.3 Risk Level	Medium	High	Low	
2.0 Sustainable Periodic Timber Harvests	2.1 Balanced Size Classes	2.1.1 Regeneration = 5-10% Forest Area	25%	27%	23%	
		2.1.2 Sapling/Pole = 35 - 45% Forest Area	32%	12%	24%	
		2.1.3 Small Sawv. = 25-35% Forest Area	43%	48%	43%	
		2.1.4 Large Sawv. = 10-15% Forest Area	0%	15%	10%	
	2.2 Full Stocking	2.2.1 Pine Plantations >= 90%	Y	Y	Y	
		2.2.2 Mixed Pine - HDW >= 80%	N/A	N	Y	
		2.2.3 Hardwood >= 70%	Y	N	Y	
3.0 Bobwhite Quail	3.1 Favorable Landscape	3.1.1 Scattered Small Fields = 20 - 25% Total Arbes	5%	25%	15%	
		3.1.2 Regenerate 25% Forest Area/10-12 yrs	25%	27%	23%	
	3.2 Food Availability	3.2.1 Hexagonal Food Plots >= 10	0	20	5	
	3.3 Cover Requirements	3.3.2 Mast Producing Oak Forests = present	N	Y	Y	
		3.3.1 Brush Piles & Hedges Near Food = present	N	Y	N	
		3.3.2 Riparian Dense Shrub Cover = present	N	Y	Y	
4.0 Wild Turkey	4.1 Balanced Size Classes	4.1.1 Regeneration = 5-10% Forest Area	25%	27%	23%	
		4.1.2 Sapling/Pole = 35 - 45% Forest Area	32%	12%	24%	
		4.1.3 Small Sawv. = 25-35% Forest Area	43%	48%	43%	
		4.1.4 Large Sawv. = 10-15% Forest Area	0%	15%	10%	
	4.2 Favorable Landscape Pattern	4.2.1 Park-like/Large HdW Near Fields = present	N	Y	N	
		4.2.2 Scattered Small Fields > 10% area	N	Y	Y	
		4.2.3 Old/Open/Large Pine = 10% area	0%	10%	0%	
		4.2.4 Park-like/Open Forest = 80% area	90%	80%	50%	
	4.3 Food Availability	4.3.1 Hexagonal Food Plots >= 10	0	20	5	
		4.3.2 Mast Producing Oak Forests = present	N	Y	Y	
	5.0 Whitetailed Deer	5.1 Favorable Landscape	5.1.1 Forest Land = 80 - 90% area	90%	80%	80%
			5.1.2 Grassland/Cultivated Land = 10 - 20% area	5%	10%	10%
			5.1.3 Brushland/Forest Openings = 10 - 20% area	5%	10%	10%
		5.2 Food Availability	5.2.1 Park-like/Large HDW w/Mast = exist	N	Y	N
5.2.2 Cultivated Corn/Grain Fields = exist			N	Y	Y	
5.3 Cover Requirements		5.3.1 Riparian Dense Shrub Cover = present	N	Y	Y	
		5.3.2 Dense Forest Understory = exists	N	N	Y	

Figure 2: By comparing the outcomes of different management scenarios, you can choose the right management plan to meet your goals.

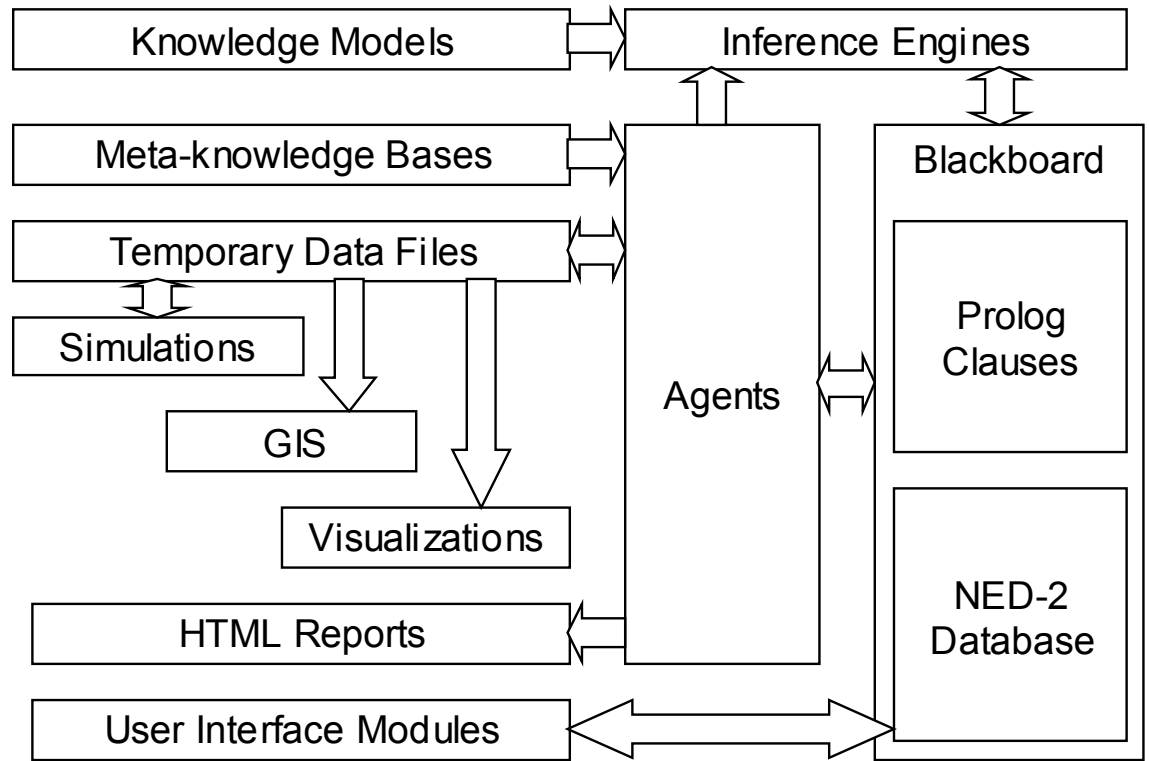


Figure 3: The NED-2 architecture

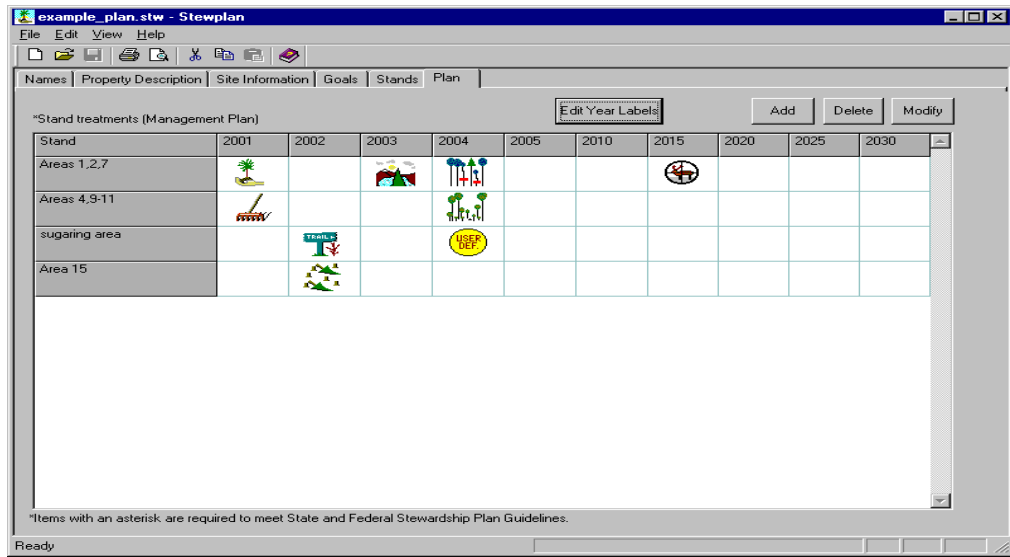


Figure 4: The NED management plan screen

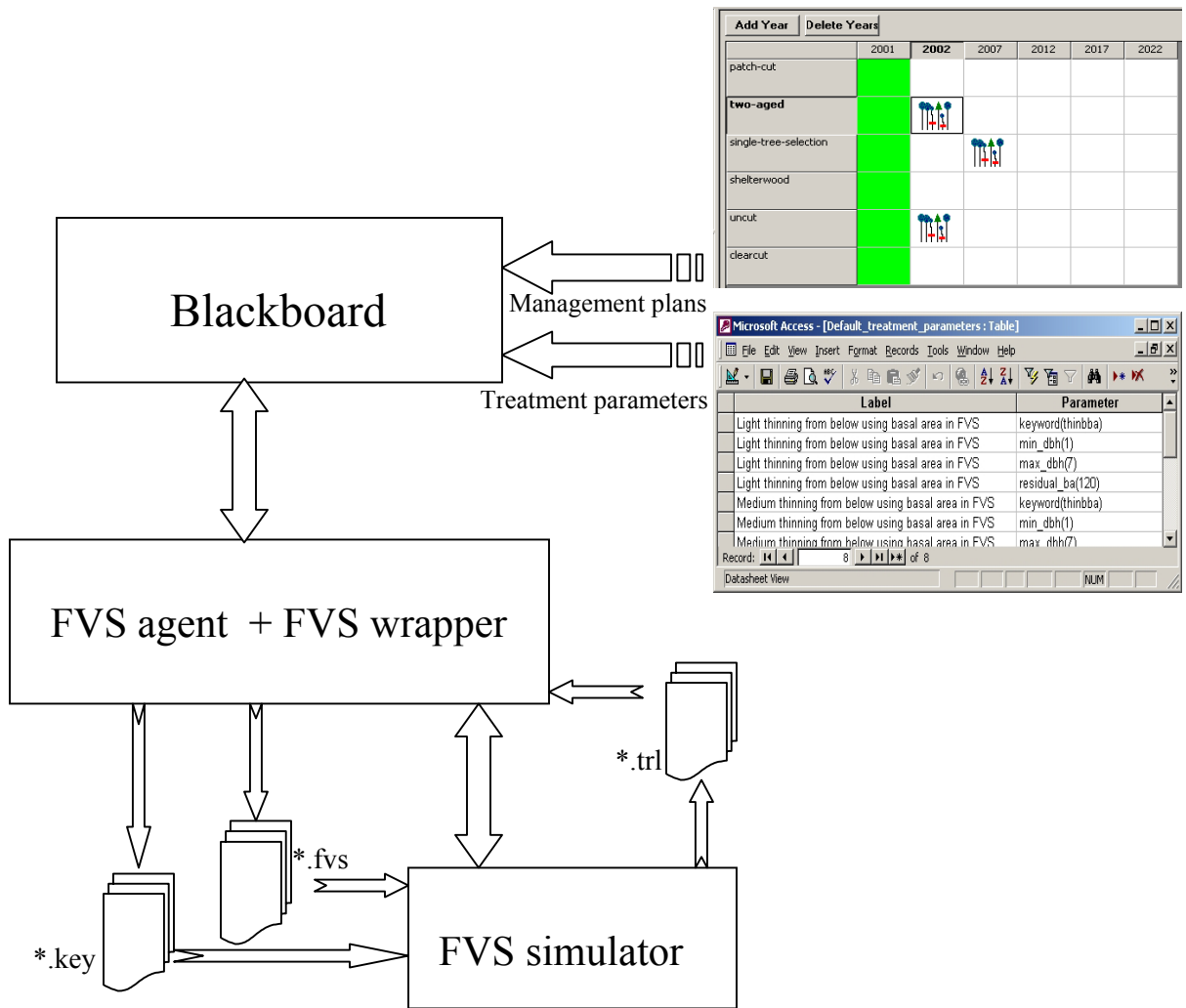


Figure 5: NED/FVS integration