AN EXPERT SYSTEM INTERFACE FOR THE GRASS GEOGRAPHIC INFORMATION SYSTEM

Katarina Johnsson[1], David G. Goodenough[2], and Cornelius Kushigbor[3]

Canada Centre for Remote Sensing
Ottawa, Ontario, Canada K1A 0Y7

ABSTRACT:
An interface is being developed to control the GRASS software (Geographic Resources Analysis Support System of the US Army Corps of Engineers) from within an expert system, as part of the SEIDAM (System of Experts for Intelligent Data Management) Project. The interface will connect to RESHELL, the PROLOG-based expert system shell at CCRS. At a low level the interface parses GRASS commands, executes them and - when needed - interprets and translates the output from GRASS to a form that other expert systems within SEIDAM can use. At a higher level, expert systems can be created that translate user-defined tasks into sequences of GRASS commands and execute those commands. Expert systems have been created for data import, data export, format conversions, data display and processing of digital elevation data. The functionality of the expert system interface is demonstrated using data from forest test sites near Invermere, British Columbia.

KEY WORDS: Knowledge based systems, GIS, forest applications

1. INTRODUCTION

In cooperation with the U.S. National Aeronautics and Space Administration (NASA), the Canada Centre for Remote Sensing (CCRS) and the Pacific Forestry Centre (PFC) are pursuing a joint project, System of Experts for Intelligent Data Management (SEIDAM). The objective of SEIDAM is to create a system of experts for intelligent data management which will integrate remote sensing data from satellites and aircraft with geographic information systems and manage large archives of remotely sensed data for dynamic selection of data sources and sensor characteristics for recognition of forest objects appropriate for environmental forest monitoring in response to user queries. SEIDAM involves technologies such as remote sensing image analysis, geographic information systems, modelling, artificial intelligence, and multimedia for explanation. Queries from the forestry domain will be used to demonstrate the capabilities of SEIDAM.

SEIDAM builds on previous projects at CCRS, in particular on the project entitled System of Hierarchical Experts for Resource Inventories (SHERI). The first two objectives of SHERI are to develop multiple expert systems in order to update forest geographic information files (maps) and to perform quality control of existing forest inventory information using Thematic Mapper imagery.

The expert systems are implemented in RESHELL, a PROLOG based expert system shell, developed at CCRS [Goodenough et. al., 1987]. RESHELL allows for a network of communicating expert systems. The breadth of knowledge in SEIDAM ranges from general semantic rules at the higher levels to task-oriented data manipulation experts at the lowest level. Each data domain (e.g. field measurements, remote sensing data and GIS data) has its own set of expert systems, that communicate through their common higher level expert (Figure 1). The lowest level of expert systems for each of the data domains in the expert system hierarchy consists of procedural and domain specific experts. The purpose of these experts is to call external software, such as GIS or image analysis software - to do the analyses. SEIDAM utilizes existing software as much as possible. For the remote sensing image analysis, LDIAS is used. LDIAS is a software package for image analysis of satellite and aircraft data developed at CCRS [Goodenough, 1988]. For the purposes of this study, the Geographic Resources Analysis Support System (GRASS), developed by the US Army Corps of Engineers, will provide the analysis capabilities for the GIS domain.

Previous Interactive LDIAS Task Interfaces (ILTI) have been developed to enable communication between expert systems developed in RESHELL and LDIAS modules [Robson et.al., 1990]. The ILTI permits the expert systems to run on one computer while controlling distributed devices and processes on other computers. A similar interface is needed to make the GRASS functions available to the expert systems within SEIDAM. The objective of this paper is to describe the RESHELL-GRASS interface, and to show how expert system technology can bring intelligence into GIS analyses and related data management.

[1] Dept. of Photogrammetry
Royal Institute of Technology
S-100 44 Stockholm, Sweden

[2] Pacific Forestry Centre, Forestry Canada
506 West Burnside Road
Victoria, BC, Canada

[3] Intera Information Technologies Ltd
2 Gurdwara Road
Nepean, Ontario, Canada

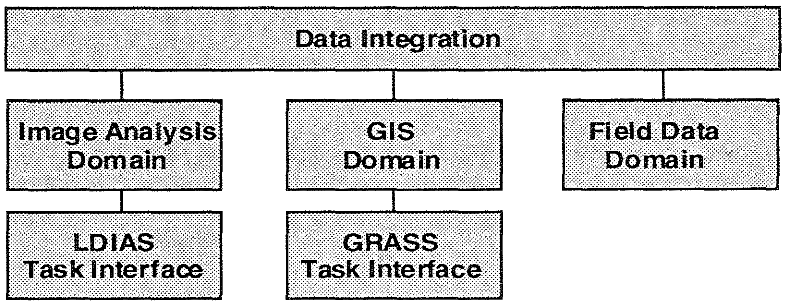

Figure 1. Each data domain has its own set of expert systems, that communicate through their common higher level expert.

2. SOFTWARE AND DATA

2.1 The GRASS Geographic Information System

Most of the analysis functions in GRASS are raster based, but the system has both raster and vector capabilities, as well as routines for conversion between the two formats [Westervelt, 1991]. In addition, GRASS includes image analysis functions and tools for graphics production (Table 1). GRASS is public domain software and it has been, for the most part, developed by researchers at the US Army Corps of Engineer's Construction Engineering Research Laboratory (CERL). There are more than 2,000 sites using GRASS currently.

TABLE 1. The GRASS programs, grouped according to the kind of tasks they are designed to perform.

Program Identifier	Description
g.xxx	general data management
d.xxx	display
r.xxx	raster manipulation
v.xxx	vector manipulation
i.xxx	image analysis
s.xxx	point (site) analysis
m.xxx	miscellaneous

GRASS is written in C and UNIX for UNIX-based computers. The latest version of GRASS (GRASS 4.0) can in fact be considered to be an extension to the UNIX operating system [Westervelt, 1991]. GRASS programs can be executed from the command line, supplying parameters as flags and options, as in the following example:

```
r.surf.idw [-e] input=name output=name
[npoints=value]

(r.surf.idw - Surface interpolation utility for
raster map layers.)
```

The GRASS data base is built upon the UNIX directory structure and the data can be accessed through UNIX commands as well as GRASS programs. Initializing GRASS is equivalent to setting a number of UNIX environment variables. The X-Window interface is used for the graphics display of maps and images.

2.2 The RESHELL expert system shell

The Remote Sensing Expert System Shell (RESHELL) is written in Quintus PROLOG. RESHELL is designed to support hierarchical, communicating expert systems in a distributed computer environment. Knowledge and facts are represented in several forms. A frames data base is used for long term storage of knowledge and for communication between the expert systems (Figure 2).

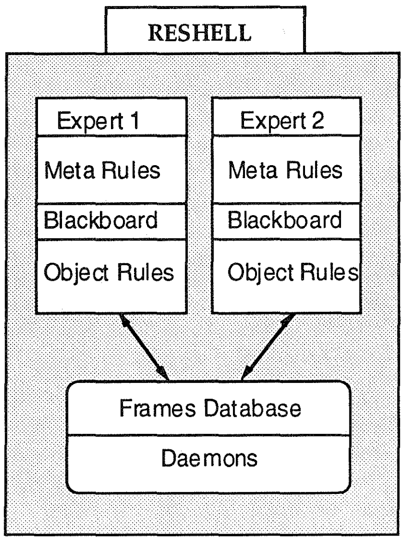

Figure 2. The experts in RESHELL communicate through the frames database.

One frame or a family of frames is designed for each of the tasks that are to be performed in SEIDAM:

```
frame_name: raster_image_to_be_displayed

slot_list: [image_name, value, "land_use"]
```

Short term knowledge is captured in two sets of rules, meta rules and object rules. These can be evaluated by backward or forward chaining. Object rules make deductions on domain objects; i.e. on the elements (maps, images or certain geographic regions) to be manipulated by the expert systems. An object rule is on the form

```
IF [<list_of_conditions>]
THEN [<list_of_actions>],
<rule_number>,
<certainty_factor>,
```

where the list of actions may include calls to external programs to carry out the tasks. Meta rules define the control strategies. Their structure is similar to the structure of the object rules. However the actions of the meta rules control process flow, which may be calls to object rules or lower level experts. Uncertainty values can be assigned to frames, objects or rules. RESHELL makes use of a graphical users interface following the MOTIF window standard.

2.3 Test data

The test data set covers two digital 1:20.000 forest map files near Invermere in southeastern British Columbia, Canada. These B.C. Ministry of Forest maps can have polygons with up to 100 attributes. The data set also includes geocoded LANDSAT TM satellite data and Digital Elevation Model (DEM) data.

3. METHODOLOGY

3.1 Requirements for the interface

The purpose of the GRASS Interface is to enable execution of GRASS programs from expert systems within SEIDAM. The first requirement, therefore, is to parse the components of the commands (name, flags and options) and to execute the commands. The interface must also have capabilities to create input files for some of the GRASS programs (e.g. for *r.combine*, a raster overlay function) and to interpret the output files from other GRASS programs (e.g. from *r.report*, a statistics report function). In so doing, the interface communicates information between a calling expert system and GRASS. This data communication should take place through the frames structure of RESHELL.

The integration of remote sensing and GIS (and potentially other analysis methods) is done on a high level in the SEIDAM expert system hierarchy. On a low level we still have two separate systems for data analysis. Data have to be imported and exported to and from GRASS. The import and export of data may involve format conversions and in some cases geodetic transformations (e.g from one coordinate system to another). The interface should support conversion and transformation tasks in a transparent fashion, by supplying tools that are customized for the types of data that will be used in SEIDAM.

3.2 Implementation of the interface

Just like RESHELL itself, the interface to GRASS is being implemented as PROLOG predicates. In pseudo code, a PROLOG predicate to display an image in GRASS would look like:

```
grass_display_image :-
      parse_the_GRASS_command_
                    with_the_correct_image_name,
      execute_the_GRASS_command.
```

The predicate has two conditions. At run time the system attempts to prove that the predicate is true by testing if its conditions are true.

Because of the close link between GRASS and UNIX, the GRASS programs can be executed as if they were UNIX commands. Quintus PROLOG provides a standard function to access UNIX commands; namely, the library predicate:

```
unix(shell(COMMAND))
```

where COMMAND is a UNIX command; e. g. *unix(shell(ls -l *))* to list all files in the current directory. Thus, the actual PROLOG predicate to display an image will be:

```
grass_display_raster_image(IMAGE) :-
       concatenate_list(['d.rast map=', IMAGE],
                                          COMMAND),
       unix(shell(COMMAND)).
```

At runtime, the two conditions become true if they are successfully executed. The *concatenate_list* is a predicate - a part of RESHELL. It parses elements in a list to a string and assigns the string to the variable COMMAND.

The *grass_display_raster_image* predicate is an example of the lowest level of the hierarchy of PROLOG predicates in the interface. These low level predicates can take one or more parameters as input, parse commands and send them to the UNIX shell. The predicates get their parameter(s) from the task frames, where they have been put by other expert systems or through user input. Other RESHELL predicates provide access to the frames and can be used either on the interface level, to build higher level predicates that include reading from one of the task frames, or on an expert system level, to create an object rule that reads from the frames and displays the image. An example of a created object rule is:

```
IF
       task_is_display_an_image_in_GRASS
THEN
       frame_get(task_display_image, image_name,
                                             IMAGE),
       grass_display_raster_image(IMAGE).
```

It is a design consideration whether to implement the whole operation, including the frame_get, on an interface level or on an expert system level. The advantage of rule creation is that the knowledge is explicitly available to the user for subsequent changes.

A number of the GRASS programs require more interaction than a simple set of parameters. Some of the analysis programs (e.g. *r.combine, r.infer*) need instructions for how to combine the data. These programs are designed to read instructions from an input file and carry out their tasks accordingly. Other programs are designed to write various kinds of statistics for the data layers to output files. Both types of programs require sets of PROLOG predicates and RESHELL rules that can interact with their multiple input or output files.

4. RESULTS

Currently we have developed approximately 100 PROLOG predicates plus RESHELL rules, mainly for the GRASS display and raster analysis commands. Currently it is possible to interface 35 programs of the approximately 200 programs in GRASS. A smaller number of higher level predicates have been developed for tasks such as display of data and creation of input files to some GRASS modules (e. g. *r.combine*). In addition, an expert system for DEM processing is under development.

4.1 Intelligent Data Display

The process of displaying an image involves GRASS system complexities, such as the need to first create a graphics window; i.e. a graphics monitor. Within GRASS, a limited number of monitors (7) with equal characteristics are available. Every monitor has a name and the user has to know the names (or know how to find them) before she / he can start a monitor. She / he might also want to check that the monitor is not currently being used by another application or another user, in a multi-user environment. The simple task to display an image, e.g. the result of an analysis, has become one that involves a number of steps and decisions (fig. 3). When one adds to this display need, the need to specify the best channels and features, it becomes apparent that expert knowledge acquisition is required to simplify the operation for the less experienced user.

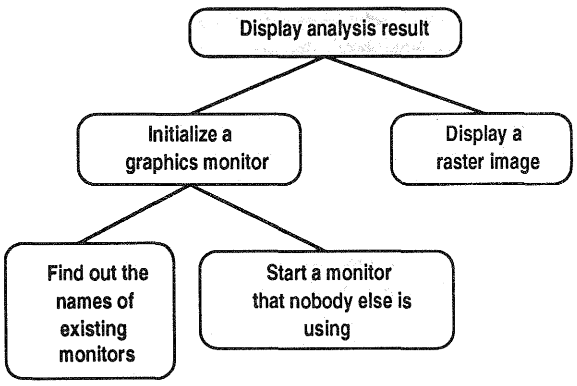
Display analysis result
Initialize a graphics monitor
Display a raster image
Find out the names of existing monitors
Start a monitor that nobody else is using

Figure 3. The first steps required to display an image in GRASS.

Two high level PROLOG predicates have been defined to display images, using GRASS:

```
/* Display an image, one argument */
grass_display_map(IMAGE) :-
      grass_init_graphics(MONITOR)
      grass_display_raster_image(IMAGE)

/* Display an image, two arguments */
grass_display_map(IMAGE, MONITOR) :-
      grass_init_graphics(MONITOR)
      grass_display_raster_image(IMAGE)
```

The first predicate will select a monitor without user interaction, while the second predicate allows the user to specify a particular monitor. An expert system should not limit the expert-users possibilities to run the program, but only help and simplify the operation for users less familiar with GRASS. These predicates are an example of this dual purpose by including the possibility to explicitly specify a monitor, thus giving the same functionality through the interface, as when the user is running GRASS directly. The predicates trigger the evaluation of a set of PROLOG predicates, that interact with three GRASS programs to extract the status of the monitors and to initialize the appropriate monitor for display (table 2).

Table 2 Tasks and the corresponding GRASS commands

Task	GRASS command
Find out the names of existing monitors.	d.mon -L
Select a monitor that nobody is using, by reading from the list.	-
Start the monitor A graphics window is created.	d.mon start=monitor_name
Display the image	d.rast map=image_name

The GRASS interaction includes interpreting an output file (from *d.mon -L*) and acting according to the following rules: *(1) The user has not specified a particular monitor and free monitors are available.* Select one of those free monitors to display the data. *(2) No free monitors are available.* Basically, if all monitors are used, the user cannot display the data. However, it happens quite frequently that the user neglects to close monitors after a task is finished. Therefore, if any of the running monitors are connected to processes owned by the same user, one of these can be selected, possibly after checking with the user. *(3) The user has specified a particular monitor.* Check that this monitor is not being used by someone else and select it.

GRASS supplies the tools to display, analyze and manipulate geographically referenced data. The tools are quite powerful, but they have to be used in a meaningful manner. The knowledge that is implicitly required to run the programs can be made explicit by implementing analysis strategies as rules in RESHELL-based expert systems. Displaying a satellite image includes selecting the appropriate band or combination of bands for the given task. Additionally, the images need enhancement and the color tables have to be set properly. Using the RESHELL-GRASS interface predicates, rules for image display can be implemented in an expert system:

```
IF
      task_is_display_land_water
THEN
      task_is_select_one_image_band_land_water
      task_is_display_an_image_in_grass.
(rule  1)

IF
      task_is_select_one_image_band_land_water
THEN
      select_TM_four(IMAGE),
      grass_histogram_enhancement(IMAGE),
      grass_color_grey(IMAGE),
      frame_put(task_display_image, image_name,
IMAGE).
(rule 2)

IF
      task_is_display_an_image_in_grass
THEN
      frame_get(task_display_image, image_name,
IMAGE),
      grass_display_raster_image(IMAGE).
(rule 3)
```

4.2 DEM Processing

Digital Elevation Models (DEMs) are crucial to a number of analysis and modelling tasks. A DEM can be used to produce slope and aspect images, as for a Digital Terrain Model (DTM), to create 3-D models of the terrain, and to perform radiometric and / or geometric correction of satellite data. The terrain is one of the most important factors for the distribution of soil types, water content, nutrient content and other factors that determines the type of ecosystem. Thus, for ecological modelling and simulations a DEM is one of the necessary input data layers and it can be used to estimate a number of other parameters. For these reasons, expert systems for various kinds of DEM processing, have a high priority in SEIDAM.

The expert system for conversion and interpolation of point elevation data to a surface covering DEM for a particular application will serve as an example of how knowledge can be distributed through out a hierarchy of expert systems (Figure 4). The higher level experts describe the tasks in broader terms but do not contain specific knowledge about how to execute these tasks. The expert systems on the lowest level are task specific and they only "know" how to execute one particular analysis task, nothing about the context of the analysis. In this way knowledge is distributed over the network, allowing each expert system to maintain a narrow focus. This approach also facilitates the design and development of the system for multiple applications. The network of expert systems is flexible in that it can be altered dynamically an automatically depending upon the input data and the success in achieving the user's goals.

The Digital Elevation Data for our data set are delivered as ASCII point files (TRIM data) and must be converted to raster format and interpolated to form a surface data layer for subsequent use. One of our applications is geometric correction of satellite images in mountainous terrain, like our test areas. The geometric correction is performed by the LDIAS software and so the DEM has to be delivered in the correct format.

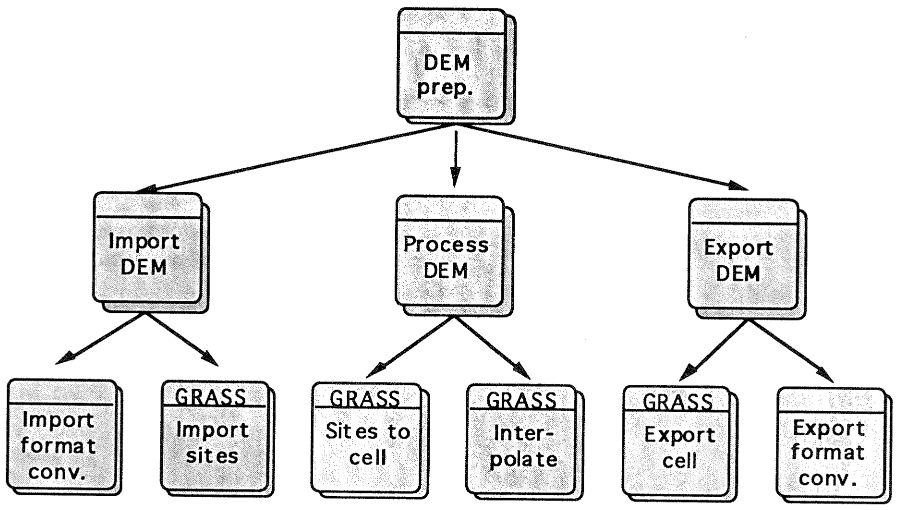

Figure 4. An expert system hierarchy for DEM processing.

The *DEM Preparation Expert System* is a higher level expert system whose task is to deliver a surface-covering raster DEM for the image analysis domain (figure 4). The scope of the expert system is set by its task definition and by the allowed input and output. The expert is called, with a set of data, from a higher level expert. The DEM preparation expert has to decide on the quality of the supplied data and determine whether it can achieve the desired goal. For example, there may be a need to assess the geometric accuracy of the corrected image and compare this accuracy to the accuracy standards. When a mix of DEM sources and images are used, it is essential to put the integrated data on a common datum, such as NAD'83. The DEM expert system breaks the task down into subtasks; i.e. import, processing and export of data, and assigns the subtasks to available experts. On the lowest level of the expert system hierarchy is the RESHELL interface to the GRASS software.

5. DISCUSSION

It is a relatively straight forward task to develop the PROLOG predicates to parse and execute GRASS programs and thus make the GRASS programs available from within RESHELL. To go one step further and bring intelligence into the system, knowledge about how to run the programs must be implemented in the form of higher level expert system rules. This study has demonstrated some capabilities of the interface and of RESHELL-based expert systems for display and analysis with DEMs using GRASS programs. Continuing research will address intelligent data analysis by integration of data from various sources in a GIS. It will focus on questions related to accuracy and to the varying spectral and spatial resolution of the data so that responses to queries in SEIDAM will contain accuracy ranges for predictions, such as quantitative estimates of forest growth.

The non-interactive command line interface is a new feature for GRASS 4.0. Our experience from developing the RESHELL-GRASS interface support the concept of this simple command line interface. It has simplified the efforts needed for the RESHELL-GRASS interface development. Even in this version of GRASS, the command line interface has not yet been fully implemented for all programs. We have encountered some functions that still require interaction. RESHELL can also execute interactive programs, but such interactions can complicate the interface. As GRASS is available with the source code and with an extensive library of functions, to modify GRASS programs is not a major problem. Future versions of GRASS are expected to have command line interfaces to all programs.

6. CONCLUSIONS

The RESHELL-GRASS interface makes GRASS programs available to calling expert systems and simplifies the execution of the programs by reducing GRASS system complexities. The interface is being developed as PROLOG predicates. Currently, 35 programs out of the approximately 200 GRASS programs can be interfaced from RESHELL. The RESHELL-GRASS interface makes it possible to develop expert systems for intelligent GIS analyses and it will form part of the research environment for the SEIDAM project, especially for spatial modelling.

7. ACKNOWLEDGEMENTS

The research and development described in this paper has been carried out at the Canada Centre for Remote Sensing (CCRS) in Ottawa. Thanks to all members of the SHERI team for valuable discussions. The forest cover data have been provided by the Inventory Branch, British Columbia Ministry of Forests, Victoria, BC. The financial support from the Swedish Institute and the Sweden-America Foundation for one of us (KJ) is gratefully acknowledged.

8. LITERATURE

Goodenough, D., Goldberg, M., Plunkett, G. and Zelek, J. 1987: An expert system form remote sensing. IEEE Geoscience and Remote Sensing, Vol. GE-25, No. 3. pp. 349-359.

Goodenough, D. 1988: Thematic Mapper and SPOT Integration with a Geographic Information System. Photogrammetric Engineering and Remote Sensing, Vol. 54, No. 2. pp. 167-176.

Robson, M., Goodenough, G., Deguise, J. 1990: Automated Program Execution in a Hierarchical Expert System: RESHELL. Proceedings of the 23rd DECUS Canada Symposium.

Westervelt, J. 1991: Introduction to GRASS 4. U.S Army CERL, Champaign, Illinois. 18 pages.