VGIS — a Graphical Front-End for User-Oriented Analytical GIS Operations

Jochen Albrecht, Hartmut Brösamle and Manfred Ehlers
[jalbrecht, ehlers]@ispa.uni-osnabrueck.de
ISPA, University of Vechta, P.O. Box 1553
D-49364 Vechta, Germany

Intercommission III/IV

KEY WORDS: GIS, Tasks, User, Integration, Experience, Process Modeling, Systems, Theory

ABSTRACT

The ever increasing functionality of GIS makes these systems more and more complex. Goal of the Virtual GIS (VGIS*) graphical user interface is to facilitate the work with GIS and to allow its use for non-GIS experts as well. The user interface consists of a flow-charting environment where operation icons can be freely combined and moved around. The operations are data structure-independent; all format conversions are performed transparently (although the user may enforce explicit consent). Where necessary, the user is prompted to enter parameters such as buffer width or overlay type. The free recombination of operator icons posed to be a tricky problem that could be solved elegantly by using the WiT® image processing environment from Logical Vision Ltd., Burnaby (B.C.) where the built-in operators being replaced by the self-defined VGIS-operators. After building the flowchart and entering all the necessary parameters, the processing can be started. No intermediate results must be stored, only the final results are visualized.

For the comfortable input of parameters Motif®-Windows were developed with the commercial GUI builder X-Designer4®. The combination of GRASS, WiT® and Motif® is not an easy task, and especially the complex and chaotic structure of GRASS is posing many problems. First experiences with prototype applications in environmental modeling show that the results seem to be worth the efforts. Limitations of the current implementation are the restriction to analytical GIS operations, leaving the chores of data input, management and more sophisticated (cartographic) output to the user. Also, GRASS™ uses a rather awkward windowing scheme that impedes the development of a truly universal GIS front-end because the interpreter between the VGIS shell and the underlying GIS has to adapt to output guidelines developed in the era before X-Windows. All these constraints, however, do not affect the general proof of concept. Current work on an interpreter between VGIS and ARC/INFO© is at a too early stage to account for similar experiences as for the GRASS™ interface. Depending on the progress of the Open GIS consortium, part of the project might be discontinued in favor of an interpreter for the more universal OGIS reference model. First applications of the VGIS front-end are introductionary GIS courses for students of a post-graduate program in environmental monitoring and as a modeling environment for erosion hazard research.

KURZFASSUNG

Die beständig wachsende Anzahl von Funktionen läßt Geographische Informationssysteme immer unhandlicher werden. Ziel der graphischen Benutzeroberfläche des Virtuellen GIS (VGIS) ist es, die Arbeit mit GIS im allgemeinen zu erleichtern und somit diese Systeme auch Nicht-GIS-Experten zugänglich zu machen. Die Benutzeroberfläche basiert auf dem Konzept der Modellierung mit Flußdiagrammen, in denen die einzelnen Operationen beliebig miteinander verknüpft, bzw. an andere Stellen verschoben werden kann. Dabei beschränkt sich die Zahl der zu erlernenden Funktionen auf einen kleinen Satz universaler, datenstrukturunabhängiger analytischer GIS-Operationen; der Anwender braucht sich um Datenformate nicht zu kümmern, da diese automatisch gewandelt werden. Dort, wo das System nicht automatisch fortführen kann, werden Parameter wie Pufferbreite oder Verschneidungsart abgefragt. Nachdem der Anwender sein Flußdiagramm fertiggestellt hat und alle*notwendigen Parameter eingegeben sind, wird die eigentliche Verarbeitung gestartet. Zwischenergebnisse füllen nicht die Festplatte, da nur das Endergebnis und das Flußdiagramm gespeichert werden.

Ein erster VGIS-Prototyp für das public domain GIS GRASS ist fertiggestellt und ein weiterer für Arc/Info® in Arbeit. Sobald das Open GIS Consortium seine OGIS®-Spezifikation veröffentlicht hat, wird die Arbeit an einem abschließenden, für praktisch alle marktgängigen GIS gültigen Interpreter aufgenommen werden. Das Bildverarbeitungsprogramm WiT® wurde zur Erstellung der Flußdiagramm-Oberfläche verwendet, indem die dort vorhandenen Bildverarbeitungsoperatoren durch die universalen VGIS Operatoren ersetzt wurden. Zusätzlich wurden mit Hilfe des Oberflächenwerkzeugs X-Designer4® Motif®-Fenster entwickelt, die die komfortable Eingabe von Parametern ermöglichen. Die Kombination von GRASS, WiT® und Motif® gestaltete sich als relativ schwierig, insbesondere erwies sich die chaotische Struktur von GRASS als problematisch. Endziel der VGIS-Entwicklung ist ein tatsächlich universeller Interpreter, der auf dem OGIS Referenzmodel des Open GIS Consortiums aubaut. Erste Anwendungserfahrungen in Unterrichtsmodulen des Umweltmonitoringstudiengangs an der Hochschule Vechta und im Bereich der Umweltmodellierung sprechen jedoch dafür, daß das VGIS-Projekt den Aufwand wert ist, den es bisher gekostet hat.

---

* Funding from the German Science Foundation (DFG) is gratefully acknowledged.

Since 1992, the terms Virtual GIS and VGIS have been coined and used in numerous presentations at international conferences to describe the project pursued at the University of Vechta. There is no connection to the Virtual GIS™ marketed by Erdas Corporation, Inc. (Atlanta, GA) since 1995

1 INTRODUCTION

The handling of geographical information systems (GIS) becomes increasingly difficult for the user. The complexity and sheer amount of operations offered is growing faster than the quality of user interfaces which are by far less comfortable than has become standard for other software packages such as word processors or spreadsheets. Many GIS still do not have an adequate graphical user interface, often commands still have to be entered at the systems level with a large number of parameters. An additional problem is the combination of raster and vector data. Many users are overwhelmed by the need to understand the principles underlying transformations between different data models.

Current solutions to these problems focus on fancy icon-based graphical user interfaces that depict the cryptic proprietary commands. While the GIS modules usually reflect the task flow from data input, via storage, manipulation, analysis to data output, the individual operations are data-centered. It is here that the project "Virtual Geographical Information System" (VGIS) derives its justification. The purpose of VGIS is to create a universal graphical user interface (GUI) that can be used with any current GIS. VGIS' graphical front-end builds upon a limited set of universal data structure-independent analytical GIS operations that allow the user to build processing models – transformations between different data structures are to be executed automatically. The graphical user interface works like a general purpose flow charting tool that depicts the workflow consisting of a number of processing steps to be applied to the input data. This processing plan can be created and edited interactively using a mouse. The plans can be saved just like macros, to be used later on with other data. The visual programming character facilitates easy adaptations to changing needs. A first implementation of VGIS runs on the public domain GIS GRASS™ coordinated by OGI at USACERL. Current development concentrates on writing an interpreter for the vector GIS ARC/INFO© from ESRI, Redlands, (CA), while the basic research focuses on a more universal interface to the Open Geodata Interoperability Specification (OGF 1993).
The theoretical background for the universal analytical GIS operations presented in section 2 discusses issues of data model independence, and the interfacing with the Open GIS specification. Section 3 introduces the general VGIS concept while the more technical implementation aspects are given in section 4. The second part. First applications of processing models are presented in the 5th section with a practical example given in section 6. Section 7, finally, provides an outlook on the direction of future research conducted at the Institute for Spatial Analysis and Planning (ISPA) of the University of Vechta.

2 THE UNIVERSAL GIS OPERATIONS

Almost all GIS-related research is data-centered. Even human factor studies usually take the dichotomy of raster- versus vector-GIS for granted and analyze GIS usage either on a key-stroke or an a task level (Turk 1992, Medyckyj-Scott and Hearnshaw 1993). In neither case the original purpose for using a GIS in the first place is taken into account. If a GIS is employed for assignments that go beyond mere data repository chores, then the operations *on* data become at least as important as the data themselves. Investigation of these operations in a processing model context has been neglected until very recently (Albrecht 1994, Voisard 1995, Hamilton and Worboys 1996).

Many GIS users possess expert-level knowledge in the application field in which the GIS is to be utilized, but have neither the time nor desire to learn the technical intricacies of a specific system (Albrecht 1994). The user's overall goal should not be the mastery of a new system but more productive interaction with geographic information. An obvious response is to provide a user interface which alleviates the need for specialized training. This user interface should aim at enhancing user interaction with geographic information and with geographic problem solving rather than with systems. Much of the user interface problem is therefore not a programming problem but a conceptual problem (Mark and Gould 1991). Frank (1993) illustrates the crucial importance of the *user interface* for the usability of a GIS with the following lines: "The user interface is the part of the system with which the user interacts. It is the only part directly seen and thus '*is*' the system for the user."

As knowledge has been gained on the behavior of spatial algorithms, a functional categorization has emerged (Burrough 1986, Tomlin 1990). However, it gave way to the basic architecture of GIS that every student learns (needs to learn) in the first introductionary course to GIS: data input, database management, analysis and output, hereby violating the functional continuation concept. Within each category, different means have been developed to handle the user interface. Menu and command driven operations have become the main way of interaction. Some are action driven (producing immediate feedback) while others are environment driven, having a cumulative effect that ends in an action driven function. A functional continuos GIS would be mostly environment driven.

Modern GIS software is a multidisciplinary tool that must allow for interdisciplinary support and is expected to be able to integrate a variety of different data sources. These data sources will be used in many ways and under a wide range of decision support situations. To meet these demands, a user interface is required whose generic functional model consists of a small set of universal GIS operations that allow for the automatic construction of a domain or task specific derived model. It would act as a shell based on a high level language consisting of spatial operators that have definable hierarchical constructs. These spatial operators can be organized following a programmable schema that allow them to generate the derived model. The core of such a shell, however, would be the generic functional model. Section 3 will introduce the Virtual GIS (VGIS) project which aims at implementing the above mentioned ideas.

The VGIS front-end to GRASS™ builds upon the 20 analytical GIS operations presented in (Albrecht 1996) and summarized in the following. Based on a questionnaire (Albrecht 1995) drawn from GIS operation taxonomies in the relevant literature (Aronoff 1991, Burrough 1992, Goodchild 1992, de Man 1988, Rhind and Green 1988, Unwin 1990) the functionality of GIS is analyzed from a user's pint of view.

The analysis of current user interfaces provides a good opportunity to study different approaches to the categorization of GIS functionality. Again, most operations serve non-analytical means and are therefore not of concern to this study. All systems offering special operations for particular appli-

cations, assist the user with headings such as *'terrain analysis'* or *'neighborhood'*.

In an analysis of the categorization schemes used in 'Map Algebra' and related classifications, Schenkelaars (1994) scrutinized the kind of queries that are necessary to build a spatial analytical query language. His approach can be used to differentiate between a user's and a developer's view of GIS functionality.

A large number of operations (such as sliver line removal or coordinate thinning) do not make much sense to the average user, or are regarded as a nuisance. Others, (such as 'line-of-sight' and 'viewshed analysis') are either synonymous or at least part of another and therefore confuse an occasional user. This does not mean that those with more experience should have no access to their functionality, but rather that such operations are hidden from an entry-level menu and that the system has default values for the results of each of these operations.

Generalization is a task that is closely related to *'zoom'* and scale change operations. As such, they are auxiliary and users may expect them to be performed automatically. This is not a trivial requirement and needs further research (Timpf and Frank 1995), however, the results should be hidden from the user of a GIS. In a similar vein, abstraction procedures, e.g. the reduction of an area to its centroid, or the regionalization resulting from a Dirichlet (Voronoi/Thiessen) tessellation , are tasks that keep being mentioned in the GIS literature (Aronoff 1991, Burrough 1986, Egenhofer and Frank 1992, Laurini and Thompson 1992). However, they were never mentioned by a single user of the survey. Based on the information, provided in section 1, it is fair to assume that the modularized future GIS will provide more user-friendly data input and management facilities and thereby render these functions obsolete.

Operations such as 'clump/labeling' are relics of the underlying data structure (i.e. raster-based tessellation's) and therefore need to be eliminated from the list of truly universal GIS operations. Purely geometric operations like 'line intersection' or 'point-in-polygon' are typical for the way that GIS functionality is currently implemented. These operations are too far beyond the non-technical horizon of the average users as to support them in solving tasks.

Table 1 represents a first approximation of what is left of the myriads of operations of one eliminates all those that have no direct analytical purpose. Auxiliary functions such as *'clump'* / *'labeling'* in the raster domain or *'topology building'* in the vector world have been discarded, yet it is exactly this group of operations that make up to 80% of all GIS operations in a regular session (Yuan and Albrecht 1995).

A most critical case represents all those operations that can be subsumed under interpolation and surface generation. With *'Search'* and its subsidiary *'(re-)classification'*, already one functional group was introduced that has definitely no analytic character, but is such an important predecessor to all analytic operations that it needs to be included. Likewise, interpolation is of crucial importance in some applications, while users in other domains ardently expect a GIS to perform all necessary interpolations by itself.

A number of operations carry different names in various domains, despite being essentially the same; *'cost, diffusion, spread'* is an example for such polymorphism. Two of the three 'Network' operations listed in Table 1 can be represented by other operations, i.e. the 'flow-between-regions' is easily implemented by the 'Thiessen/Voronoi' operation, while the 'shortest path' is a repeated 'nearest neighbor' operation. Figure 1 represents a conclusive list of user-oriented, analytical, universal GIS operations; mind though, that this list is not meant to be the only one that could be conceived of. Rather it mirrors the expectations of the randomly selected group of people who answered the above mentioned questionnaire (students in Austria and Germany as well as colleagues at a number of international conferences in 1993 and 1994; most of them being ARC/INFO© users).

Table 1. A first compilation of user-oriented GIS operations

- Search
  - Thematic Search
  - Search by Region
  - (Re-)Classification
- Location Analysis
  - Buffer /Thiessen
  - Corridor
  - Overlay
- Terrain Analysis
  - Slope and Aspect
  - Catchment/Basins
  - Drainage Network
  - Viewshed Analysis
- Flow Analysis and Network
  - Connectivity
  - Shortest Path
  - Flow between Regions
- Distribution / Network
  - Costs, Diffusion, Spread, Gravity Modeling
  - Change Detection
  - Proximity
  - Nearest Neighbor
- Spatial Analysis / Statistics
  - Pattern
  - Centrality
  - Complexity/Variation
  - Dispersion Measures
  - Frequency
  - Indices of Similarity/Diversity
  - Topology; Hole Description
  - Topology; Upstream Elements
  - Global Surface Fit (Trend/Fourier)
  - Multivariate Analysis
  - Regression
  - Autocorrelation
- Measurements
  - Number of Items
  - Distance
  - Direction (Calculate Bearing)
  - Perimeter, Acreage, Height, Volume
  - Surface
  - Shape
  - Fractal Dimension
  - Adjacency, Contiguity

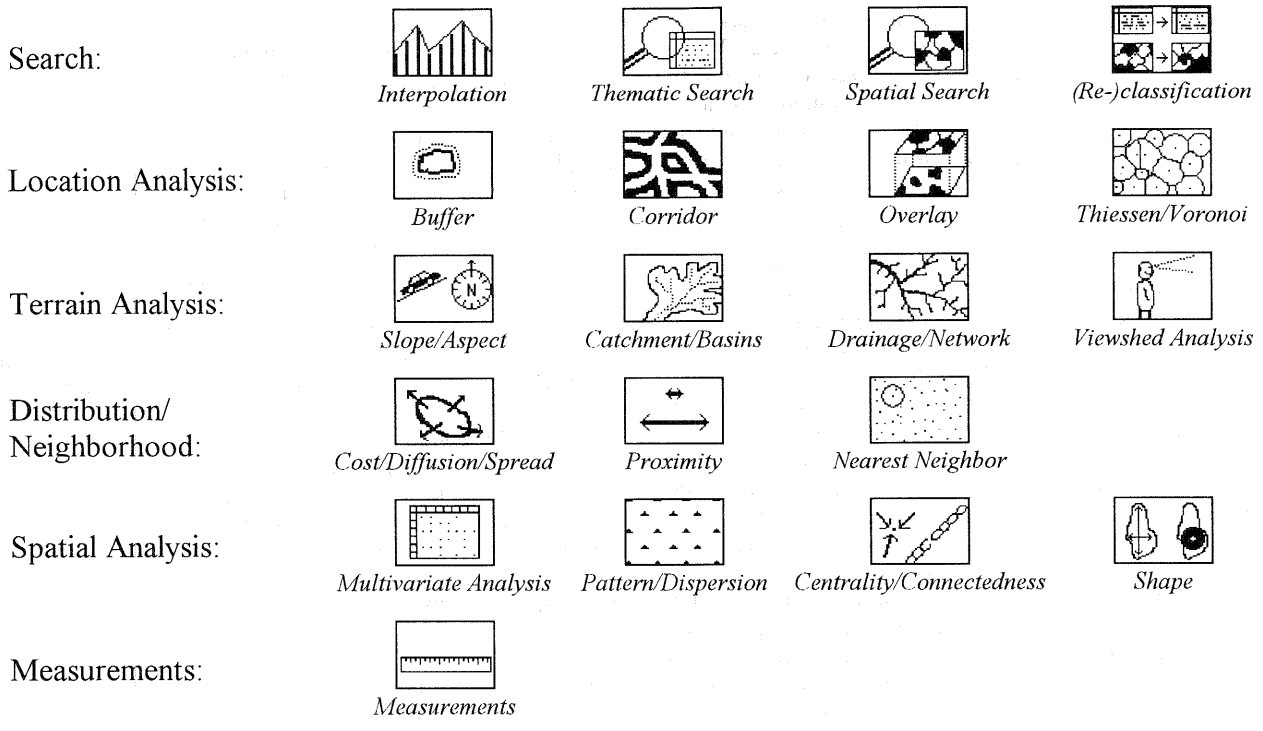

Figure 1. The 20 universal analytical GIS operations

A number of operations are well-known and there is no discussion about how to categorize them (e.g. the group of terrain analytic operations). The omission of *Network* functions is arguable; however, a GIS user interface needs to be adjusted to the field of application, and in a utility (network) application it makes a lot of sense indeed, to have network functionality as a separate group heading. In like manner, the naming of the *'cost'* operation could equally well be called *'diffusion'* or *'spread'* - it is a matter of domain, which term is more appropriate; algorithmically they are all the same.

In the following, six groups of analytical GIS operations are differentiated (group headers will always be capitalized, whereas the individual operations begin with a small letter).

Search operations can be partitioned into thematic searches and search under geometrical constraints. In most cases, the search operation is succeeded by a *'select'* and subsequent analysis of the selected object(s). As such it is not a typical analytic operation itself. It is included here, because especially the *'search-by-region'* operation can not expected to be found in other but GIS packages. The search-by-region operator uses a user-defined search (rectangular) window, an arbitrarily shaped mask, or a filter that has some spatial properties. *'(Re-) classification'* is basically a database operation. In most cases, however, the filter that is used for a *reclassification* has a spatial determinant. As a matter of fact, the whole concept of *'Map Algebra'* can be regarded as a form of reclassification. The unit of measurement does not need to be metric. Distances could be expressed just as well in time or in relative spaces, such as number of nodes in a network.

The *'Locational Analysis'* group is comprised of four operations that are among the best-known and most often used GIS operations at all. *'Buffer'* and *'corridor'* are quite similar, and it could be argued that a *'corridor'* is a *'buffer'* operation that takes two distances (the inner and the outer boundary) and can only be applied to a group of 2- or higher dimensional features. But that is a technical view of the operation and does not meet the requirement of user-orientation set forth as a prerequisite of this work.

The probably best-known analytical GIS operation is *'overlay'*. It is comprised of many other operations such as *'clip'*, *'erase'*, *'split'*, *'identity'*, *'union'* and *'intersect'*, and can be applied to any combination of spatial features. The degree of sophistication of a GIS application depends on the knowledge that a user possesses about the resultant of any of these operations. The last item in this group is the *'Thiessen/Voronoi'* operation. This one is sometimes also categorized as a *'Neighborhood'* operation. However, from a task-oriented perspective, it fits well with the three above. One operation that became completely subsumed under *'Voronoi/Thiessen'* is the *flow-between-regions*. By assigning weights to the Voronoi nodes, it is possible to simulate the flow between the Thiessen polygons. These four operations satisfy most needs in the large set of location/allocation problems. There are overlaps with functionalities in the *'Neighborhood'* and *'Terrain Analysis'* groups, but for the sake of a clear categorization with not all-encompassing groups, other operations suitable for location/allocation problems were kept with their more prototypical group headings.

The operations of the next functional group deal with explicitly or implicitly 3-dimensional data. They are all well-known and therefore need no explanation beyond the description of parameters used in the algebraic specification. *'Slope/aspect'* requires an input file that contains height values. In the exceptional case of the input file being a TIN, the result of this operation is already implicitly recorded. Although not being a truly analytical operation, *hill-shading* can be accomplished within the same operation, if the sun azimuth and the viewer's elevation are provided as additional parameters. The *'catchment/basins'* operation takes either a 'height' or a 'slope' file and calculates the extent of a single basin (if an additional selection point is provided) or the entire set of basins. Similarly, the *'drainage/network'* operation computes either the flow from a

single source location or the complete stream network. This includes the delineation of stream links, stream order, flow direction and upstream elements. The *'viewshed'* operation is the only other one that requires an input file containing height values. In addition to that a view point must be designated (this could also be a route or an area) and the viewer's height above ground needs to be specified. The search distance is an optional parameter.

The *'Distribution/Neighborhood'* group of operations is probably the most geographic of all. Non-statistical queries about the relationship between spatial features are usually answered with this set of functionalities. The *'cost/diffusion/spread'* operation takes one arbitrarily dimensioned feature and calculates the value of neighboring attributes according to some spread function. The spread can be influenced by barriers which simulate spatial impedance (the 'cost' character), accessibility, or the *relative distance* under anisotropic conditions. The spreading function is usually expressed by an equation while the friction can be represented either by an equation or by a special friction coverage. The *'shortest path'* functionality described in the *'Network'* section above, can also be implemented by a *'spread'* along a given network. Especially, if the shortest path calculation is to be based on relative distances, *'cost/diffusion/spread'* might be more appropriate than the *'nearest neighbor'* operation.

'Proximity' is less of a singular operation than a functional group of numerous technical operations that carry out the same functionality. Proximity measures can be applied to all features of an input file or to selected only. In case of multi-dimensional features, the user needs to specify whether it should be measured from edge to edge or from center to center. Finally, a maximal distance may be specified for what is considered to be proximal.

Similarly, the *'nearest-neighbor'* operation uses a number of different algorithms, depending on the mode, which usually is (but does not have to be) conditional to the input data. Aside from the common specification of input and output files, (the input can be one or several features of any type), this operation needs particulars about the unit of measurement (i.e. length or number of nodes) and the mode (e.g. along a path or as-the-crow-flies).

It could be argued that *'proximity'* belongs into the *'Measurement'* group, while *'nearest-neighbor'* is a special case of the *'cost/diffusion/spread'* operation, which in itself is nothing but a complex *'reclassification'*. This would render the whole group obsolete. From a technical point of view, this argument is valid, however, it does not correspond with the requirements on the user's side and is therefore not adhered to here.

All statistical measures, that inhibit a certain degree of complexity, are categorized as *'Spatial Analysis'*. This includes the landscape ecological *'pattern and dispersion'* measures, (such as frequency, indices of similarity, relative richness, diversity, dominance, fragmentation, density, Shannon index, and degrees of freedom) as well as *'centrality or connectedness'*, *'shape measures'* (e.g. skewness, compactness), and the whole set of tools for *'multivariate analysis'*. They all result in singular figures, which is why they could arguably be categorized as *'Measurements'*. Some of the computations, however, are so complex that users would be confused if they were grouped among measures like *'perimeter'* or *'acreage'*.

'Pattern' and *'dispersion'* measures are possibly the most prototypical of all *'Spatial Analysis'* operations, at least with respect to descriptive statistics. *'Centrality'* gives either the center of a point cluster or a measure of connectivity in a network. *'Shape'* measures are used in a wide array of applications, e.g. in geomorphological, bio-geographical, political ("gerrymandering"), or archeological practices. A number of basic parameters that can be found in the *'Measurements'* group (*'acreage'*, *'perimeter'*, *'centroid'*, etc.), are used here to describe elongation, orientation, compactness, puncturedness or fragmentation.

The last operation in the *'Spatial Analysis'* group is again a header for a whole bag of secondary operations. *'Multivariate analysis'* is comprised of a number of techniques to describe the relationships and dependencies among the spatial objects in scrutiny. Although these are definitely analytic in character, it can be argued that their functionality is covered by such well-established statistical software packages as SAS, SPSS or S-plus and therefore, do not need to be classified as a universal GIS operation. In the other hand, operations like *'regression'*, *'autocorrelation'* and *'cross-tabulation'* are so often used in a GIS context, that they are included here as well.

The *'Measurements'* group is virtually infinite. In it's core, it consists of a number of simple geometric calculations (*'distance'*, *'direction'*, *'perimeter'*, *'acreage'*, *'height'*, *'volume'*, *'surface'*, *'fractal dimension'*); these are then extended by simplest statistics (*'number'*, *'histogram'*, *'mean'*), and finish with a few topological measures, such as *'adjacency'* and *'doughnuts/holes'*.

3 THE VGIS CONCEPT

VGIS is not intended to be yet another GIS. It is conceived as a shell that can be draped over an existing GIS, using the function offered by that GIS (see Figure 2). The current implementation requires one interpreter for each existing GIS, but the prospect of vendors adhering to the OGF's Open Geodata Model alleviates this necessity.

The VGIS-Shell consists of four modules:

- the graphical user interface (GUI)
- an interpreter for the processing plans
- a tool to generate the processing plans
- the underlying of-the-shelf GIS

2.1 The Graphical User Interface

Compared to regular GIS user interfaces there are a number of significantly different features that cause the innovative user friendliness of VGIS:

- Complete reconfiguring of the user interface
  The GIS manager has complete freedom to adjust the user interface to the needs of his customers. Only those operations that are useful to solve the users' task will be offered. Their structure and presentation in sub-menus are also fully configurable.

- Powerful, complex operations
  The operations offered to the user are no isomorphic mapping of the operations of the underlying GIS. Rather they are custom-made to each application. These operations are independent of whatever data structure is used. Necessary transformations are initiated by VGIS and performed by the underlying GIS.

- Platform independence
  Since VGIS is developed independently from the underlying GIS, users may exchange workflows without need to adapt them to particular environments.

- Automatic generation of flow charts
  This feature is reserved to a future version of VGIS.

- Self-documentation through storage of processing plans that capture lineage
  Processing plans can be saved and modified anytime. The user may enhance results by iterative changes of the workflow depicted in the processing plans. Data becomes self-documenting because the processing plans contain their lineage.

Flow charts are a the standard process-oriented tool in visual programming (Chang 1990, Glinert 1988, Monmonnier 1989). Such a visual programming example is depicted in Figure 8, where the modeling flow chart allows the user to "play" with the data flow. Within VGIS, it is easy to test the result of new routing paths within the flow chart. Different hypotheses can easily be tested by adding or changing a connection of the flow chart. A similar reconfiguration of a conceptual model would require substantial GIS expertise if it were attempted in a vendor GIS.

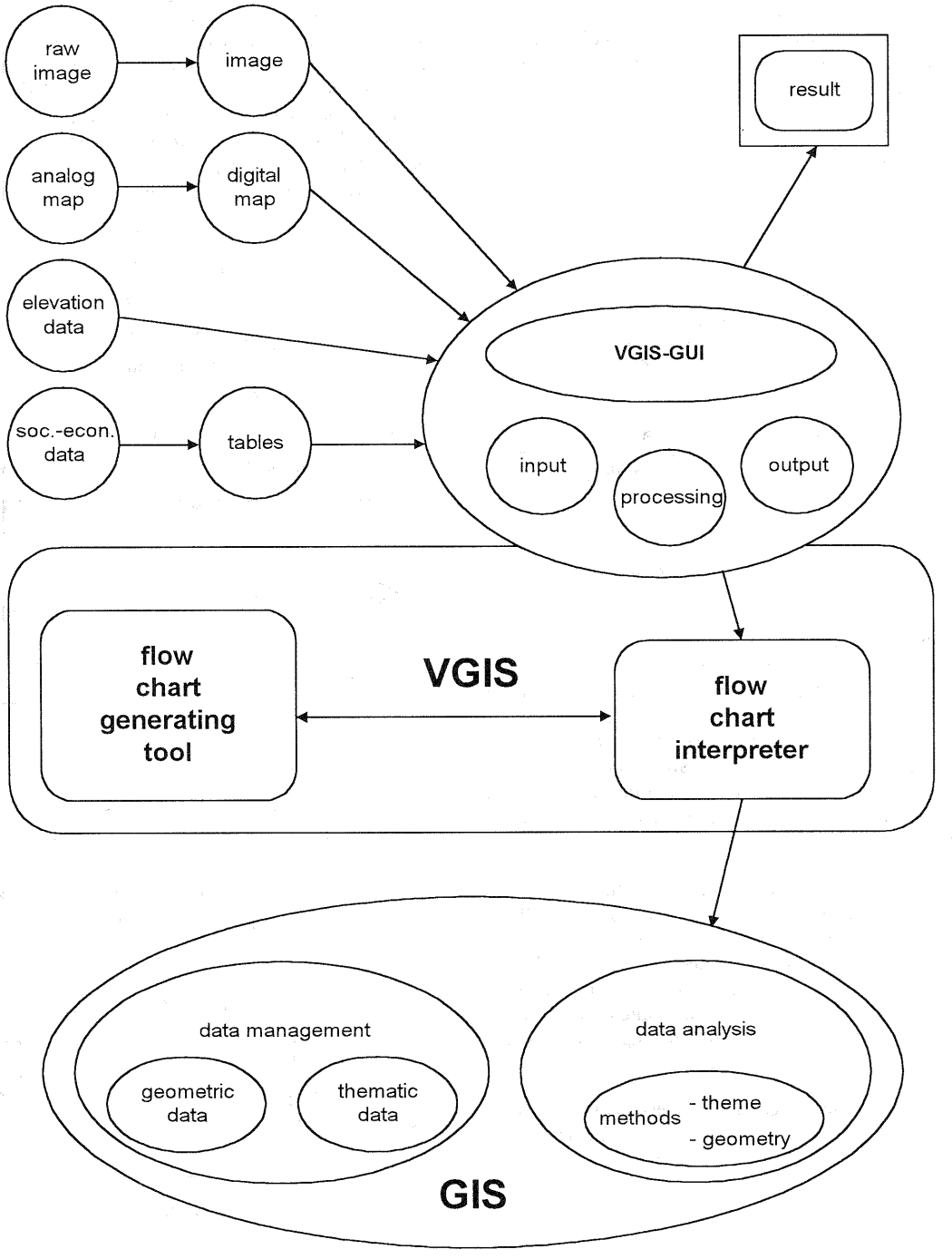

Figure 2. The VGIS design

2.2 The Flow Chart Interpreter

The interpreter used to translate processing plans into the operations of the underlying GIS consists of two steps (see Figure 3).

In a first step, the powerful and user-oriented VGIS functions are dissected into elementary GIS operations. In a second step, these elementary GIS operations are then translated into the proprietary functions of the underlying GIS.

The advantage of this two-step approach is the independence of the first phase from whatever GIS will be used in the second phase. The second step makes the actual interface between VGIS and the proprietary GIS. That way, both the custom-made user interface and processing plans can be exchanged across platforms. The more powerful VGIS functions should ideally be developed by a GIS manager who builds them from elementary GIS operations to fit a particular application. The definition of those elementary GIS operations plays a key role and is described in section 2.

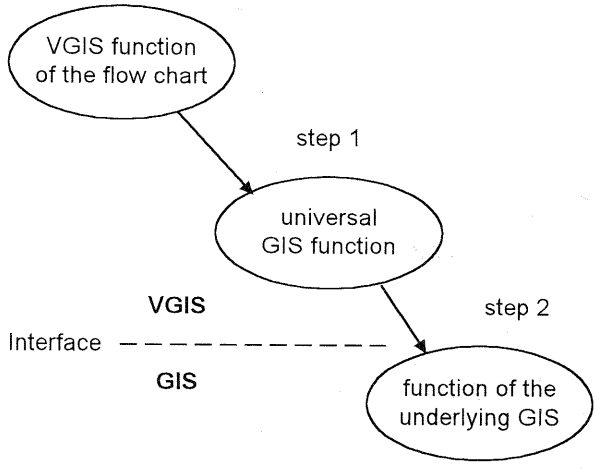

Figure 3. The two-step flow chart interpreter

2.3 The Flow Chart Building Tool

The interface between the user and the system will be further enhanced by a flow chart generating tool. Here, the user needs only to *describe* the results and to name the source data to be employed. Based on a knowledge base of integrity rules, the appropriate flow chart will then be generated automatically. An explanatory component is supposed to help the user to understand how the system derived the flow chart, which may be subsequently modified.

The flow chart generating tool is based on the principle of backward chaining. The system checks whether data with the requested attributes exists already. If it does not find such data, an appropriate procedure will be started to derive the data. In case the necessary source data cannot be found, it will recursively call other workflows until the warranted data is either produced or it will halt with an error message stating that it cannot be derived from the input specified.

The automatic generation is certainly a very complex task requiring additional funded research. It is therefore only envisaged as an extension to the current VGIS project.

4 THE VGIS PROTOTYPE

3.1 Hard- and Software

VGIS has been developed so far using a personal computer that is connected to a workstation on ethernet via an X™ server. Limited resources made it necessary to rely as much as possible on standard software under the UNIX operating system. The initial implementation uses the public domain package GRASS as GIS. A commercial program called WiT®, that has originally been developed for image processing, was used as a tool for processing plan generation. Additional functions are implemented in C and C++, while the necessary Motif®-based windows were developed with a graphical user interface builder called X-Designer®.

3.2 Design and Functionality of the Prototype

VGIS starts with a shell script that first calls a C++ program *vgis_grass_start* which allows the user to define - within a Motif® window - the environment variables required to run GRASS. Once they are set correctly, a VGIS-adapted WiT® session is started. WiT® controls the actual functionality of VGIS. Only when WiT® terminates, this control is returned to the initial shell script which then calls the program *vgis_grass_end* that deletes all temporary files and ends VGIS.

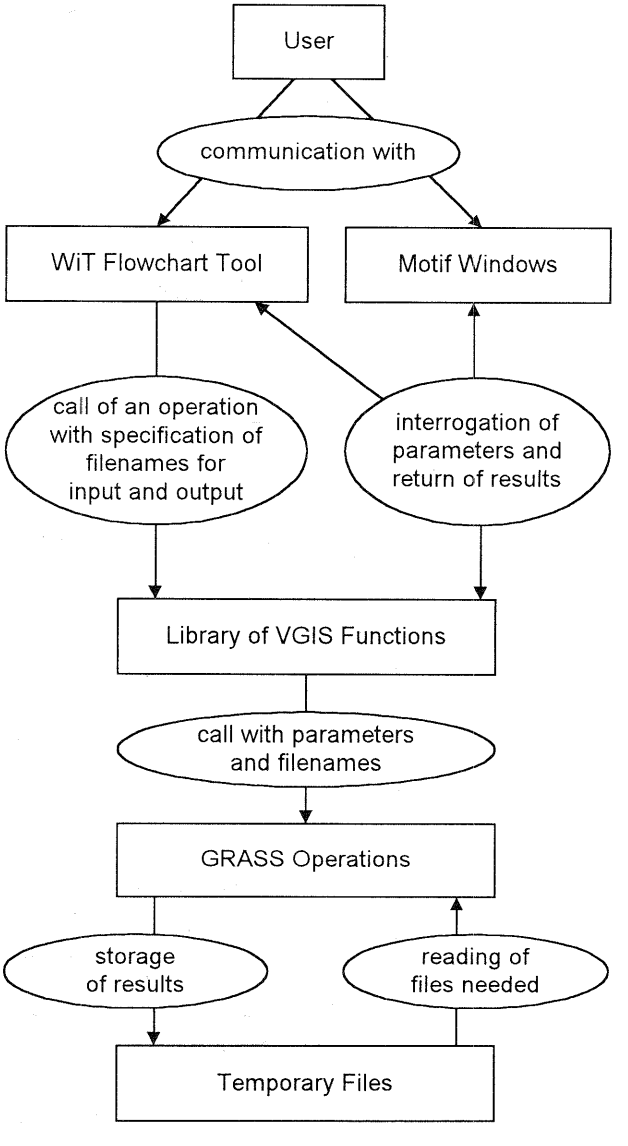

Figure 4. The construction of the VGIS prototype

Figure 4 depicts the general layout of VGIS. The user is only communicating with the WiT® user interface and self-explaining Motif® windows. The underlying GIS and all administrative tasks are completely hidden from the user. All WiT® image processing operators have been replaced by VGIS functions that are described in the following section.

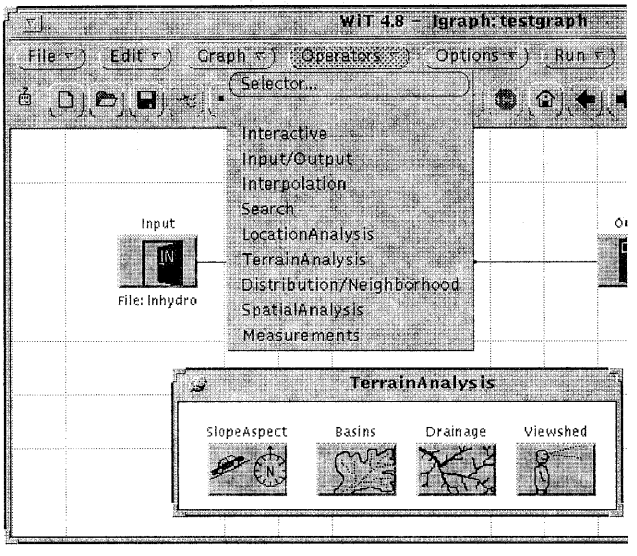

Figure 5. The modified WiT® menu

Figure 5 shows the modified WiT® menu for the selection of VGIS functions. When a menu item is selected, the respective group window opens and the actual function may then be selected (*TerrainAnalysis* in Figure 5). In addition to the eight groups of analytical GIS operations, the menu contains a group *Interactive* that supplies functions to control the processing plan.

The operations are grouped according to Albrecht (1996). He provides an extensive discussion of each operator. Figure 6 shows the operations of a number of functional group windows.

Input/Output:	Input, Output
Interpolation:	linear, Rhumbline, greatCircle, idw, potential, trend, global, Contours, Kriging
Search:	thematic, byRegion, Classification
LocationAnalysis:	Buffer, Corridor, Overlay2, Overlay3, Overlay4, Thiessen
TerrainAnalysis:	SlopeAspect, Basins, Drainage, Viewshed
Distribution/Neighborhood:	Proximity, Diffusion, Neighbor
SpatialAnalysis:	Pattern, mvAnalysis, Centrality, Shape
Measurements:	Number, Histogram, Distance, Direction, Mean, perimeter, Acreage, Height, Volume, Surface, Adjacency, Skewness, Compactness, Variation, fractalDim, Similarity, Topology

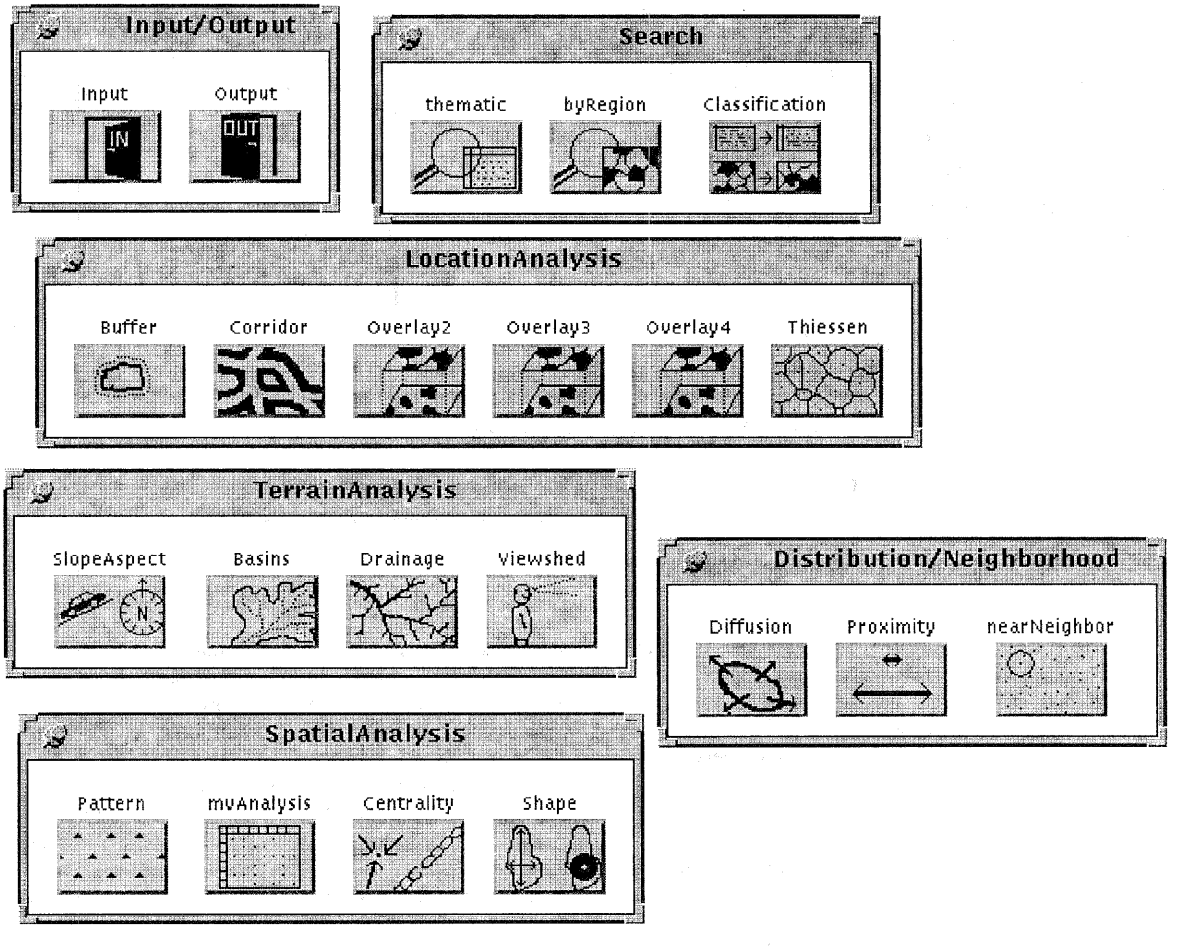

Figure 6. VGIS functions divided into thematic groups

The user outlines the workflow interactively by combining these operators to form processing plans. Selecting the WiT® menu item *Run* starts the processing of the flowchart. WiT® calls for each operator a C program with a corresponding name. Input and output filenames, as well as operator parameters if necessary, are transferred. The C program processes them in such a way that one or more GRASS commands can be issued. GRASS carries them out and saves the results on the hard disk. The names for these temporary files produced by GRASS are automatically generated. The next VGIS operator uses the temporary file as input. In case that not all necessary parameters have been provided, the C program calls a C++ module that prompts for the required data in a Motif® window. *Reclass* is an example for such an operator requiring further information from the user.

By employing several *display-* and *output* operators, intermediate results can be displayed and compared. Figure 7 depicts the previous example with the display of intermediate results.

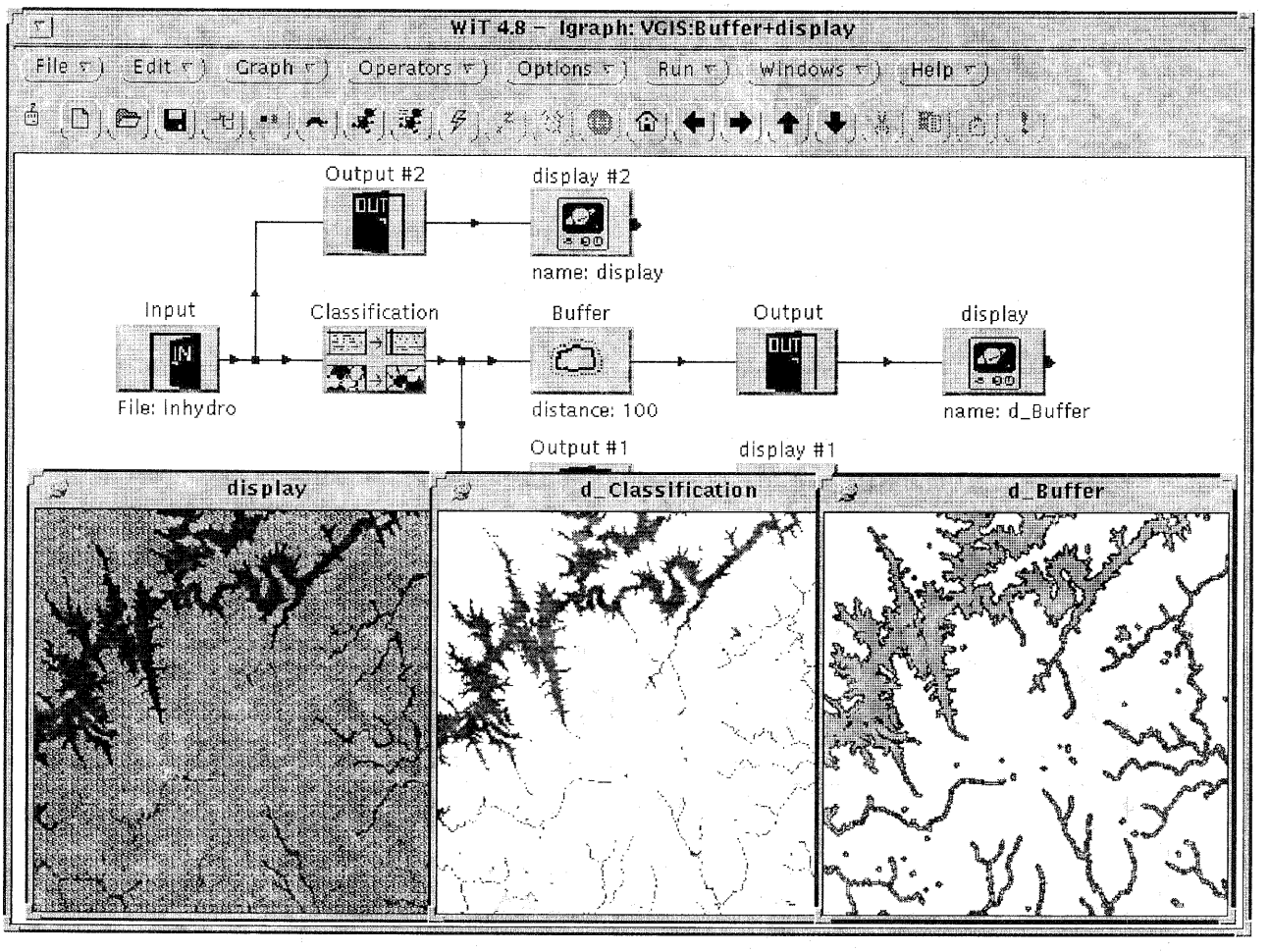

Figure 7. Display of provisional results in VGIS

5 FIRST RESULTS

User resonance to the first VGIS prototype is quite positive. The interface is easy to handle and the learning curve exceptionally low.

A few problems, mainly resulting from the use of WiT®, should nevertheless be mentioned. Direct querying of parameters in self-defined windows is not supported and can even with tricks only be accomplished while processing a flow chart. During the creation of flow charts only some very simple WiT®-provided windows can be called. This is an enormous obstacle.

In addition, it is not acceptable for a GIS application that the number of inputs of an operator needs to be fixed and cannot be parameterized. There is an unnecessary large set of *overlay* operators only because the number layers needs to be defined by the developer rather than the user. C is the only language supported by the WiT® API. In general, WiT® offers too few means for customer-driven extensions.

The probably biggest handicap of the current VGIS prototype is the lack of a real interface between its modules. Data keeps being written to disk and read from there, causing the performance to drop dramatically. The original concept of WiT® allows for images to be transferred from one operator to the next one directly. GRASS (and other GIS) however, do not accepts other input and output media than the hard disk.

The development of this prototype proved the correctness of the underlying concept. The current development of an Arc/Info® interpreter for VGIS will show that true data structure-independence is feasible. The authors came nevertheless to the conclusion that due - to the restrictions caused by employing standard software packages - some major changes to the current path of development are necessary.

6 EXAMPLE

A sample exercise used in the first semester of the post graduate course in environmental monitoring given at the University of Vechta shall illustrate the advantages of VGIS in a practical application. The task is to locate the optimal site for a factory. This task has previously been solved using the software package Erdas/Imagine®. This is then compared with the workflow within the VGIS environment.

The task is as follows:

An site for a large factory in the area of lake Lanier, Georgia is to be located, that fulfills the following six conditions:

1) Environmentalists lobbied successfully that no forested area should be used. For the same reason already built up areas are not feasible.
2) Pollution threats cause the factory to be located at least 300 m off any surface water.
3) On the other hand, the factory should be easy to access; a first order highway must be within 450 m.
4) To save taxes, the factory should be sited outside of the municipality of Gainesville.
5) A special construction requirement requires the factory to be built on Madison Sandy Loam or Madison Sandy Clay.
6) Due to the high energy requirements, the transformation station should be less than 3 km away

The following thematic maps are available for the lake Lanier area:

1) lnlandc.gis: land use classification
2) lnhydro.gis: hydrology
3) lnput.gis: political borders and infrastructure
4) lnsoils.gis: soil types

To solve this task with Erdas®, the user has to enter the following command sequences:

- *lnlandc.img* ⇒ `Recode`: forest + built up areas = 0, other = 1 ⇒ store as *file_1.img*
- *lnhydro.img* ⇒ `Search`: 300 m buffer around surface water = 10 pixel ⇒ store as *file_2.img*
- *file_2.img* ⇒ `Recode`: surface water + buffer = 0, other = 1 ⇒ *file_3.img*
- *lnput.img* ⇒ `Search`: 450 m buffer around highways type (3,4) = 30 Pixel ⇒ *file_4.img*
- *file_4.img* ⇒ `Recode`: highways type 3,4 + other = 0, street buffer = 1 ⇒ *file_5.img*
- *lnput.img* ⇒ `Recode`: urban + infrastructure = 0, other = 1 ⇒ *file_6.img*
- *lnsoils.img* ⇒ `Recode`: Madison Sandy Loam + Sandy Clay = 1, other = 0 ⇒ *file_7.img*
- *lnput.img* ⇒ `Search`: 3 km buffer around transformation stations = 100 pixel ⇒ *file_8.img*
- `Overlay (minimum value dominate)` *file_1.img* and *file_3.img* ⇒ *file_9.img*
- `Overlay (Minimum value dominate)` *file_5.img* and *file_6.img* ⇒ *file_10.img*
- `Overlay (Minimum value dominate)` *file_9.img* and *file_10.img* ⇒ *file_11.img*
- `Overlay (Minimum value dominate)` *file_11.img* and *file_7.img* ⇒ *file_12.img*
- `Overlay (Minimum value dominate)` *file_12.img* and *file_8.img* ⇒ *opt_site.img*

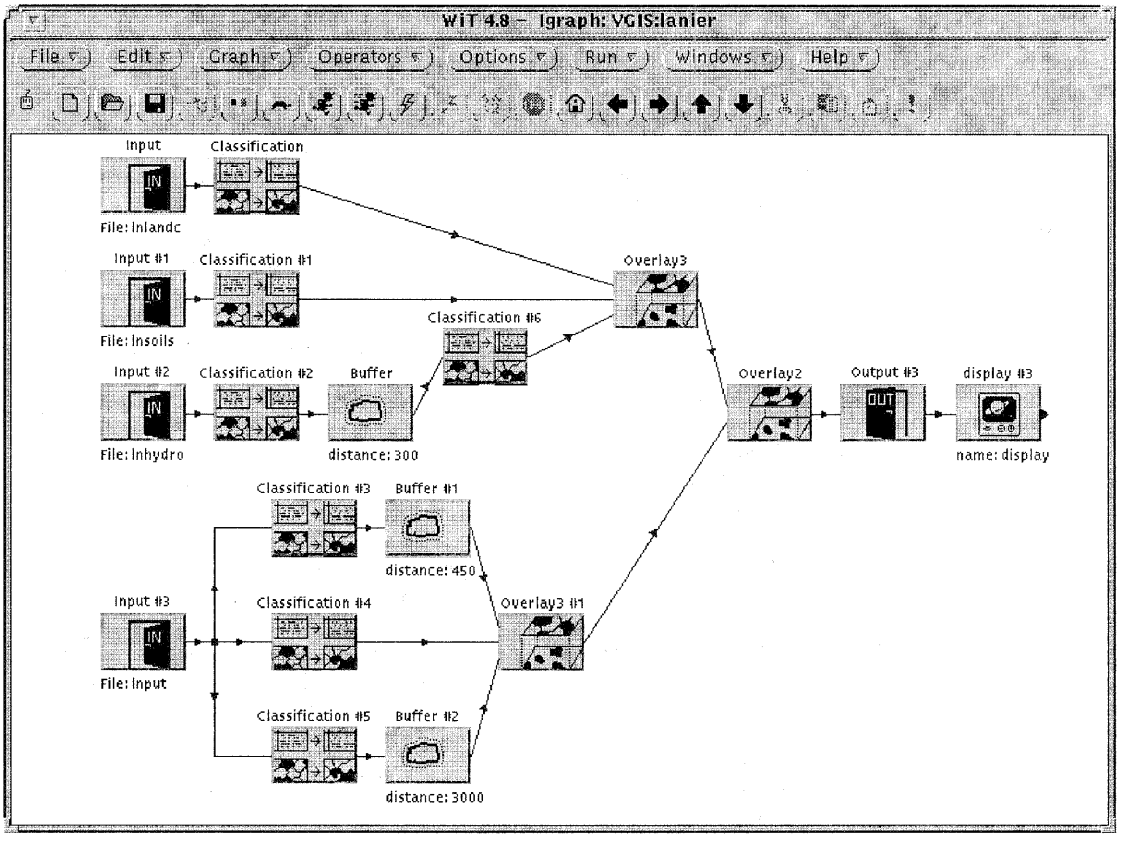

Figure 8. Solution of the site location for a factory with VGIS.

Each of those commands must be issued individually. The user is responsible for administering the names of input files and output files. The example shows that the user has to provide unique names for each intermediate result so that it can be correctly referenced in subsequent procedures. This task is a common source of errors. Commonly, users create a processing plan themselves to avoid loosing track.

Employing VGIS, the user only has to create the work flow depicted in Figure 8 and to start its execution. While processing the flow chart the user is then prompted by the system to enter the rules for reclassification interactively. The result is displayed on the screen.

7 FUTURE RESEARCH

VGIS does not yet include conditional and iteration operators as they are used in formal programming languages. The 20 universal analytical GIS operations employed do not account for any three-dimensional or temporal reasoning. The formalization given in (Albrecht 1996) can only be a preliminary one as the universal data model that underlies the definition of operations is based on an older draft of OGF's geodata model (OGF 1993) which has in the meantime been considerably altered. Finally, it would be extremely useful if not only the services-directed operations on the application level were standardized, but the low-level functions, that are actually used by OGIS-conform systems, as well. Research into this direction is currently pursued by the authors both within OGF and ISO's technical committee 211 (Geoinformation).

8 LITERATURE

Albrecht, J., 1994. Universal Elementary GIS Tasks—Beyond Low-Level Commands. In *Proceedings of the 6th Symposium of Spatial Data Handling*, pp. 209-222, Edinburgh.

Albrecht, J. 1995. Semantic net of universal elementary GIS functions. In *Proceedings ACSM/ASPRS Annual Convention and Exposition Technical Papers*, Vol. 4 (Auto-Carto 12), pp. 235-244.

Albrecht, J., 1996. Universal GIS Operations for Environmental Modeling. Proceedings of the Third International Conference/Workshop on Integrating GIS and Environmental Modeling, Santa Fe, NM. Santa Barbara: National Center for Geographic Information and Analysis (published as CD).

Albrecht, J., 1996. *Universal Analytical GIS Operations*. A Task-Oriented Systemization of Data Structure-Independent GIS Functionality Leading Towards a Geographic Modeling Language. University of Vechta, ISPA: Mitteilungen 23,.

Aronoff, S., 1991. *Geographic information systems: a management perspective*. Ottawa: WDL Publications.

Burrough, P., 1986. *Principles of geographic information systems for land resources assessment*. Oxford: Clarendon Press.

Burrough, P., 1992. Development of intelligent geographical information systems. In *International Journal of Geographical Information Systems*, **1**, 1-11.

Chang, S., 1990. *Principles of visual programming systems*. Englewood Cliffs: Prentice Hall.

Egenhofer, M. and A. Frank, 1992. Object-Oriented Modeling for GIS. *Urisa Journal* **4**(2):3-19.

Frank, A., 1993. The Use of GIS: the user interface is the system. In Medyckyj-Scott, D. and H. Hearnshaw (Eds.) *Human Factors in Geographical Information Systems*, pp. 11-12, London: Belhaven Press.

Glinert, E., (Ed.), 1988. *Visual programming environments*. Los Alamitos: IEEE Computer Society Press.

Goodchild, M., 1992a. Geographical information science. In *International Journal of Geographical Information Systems*, **1**, 31-46.

Goodchild, M., 1992b. *Spatial analysis using GIS*. 2°. Santa Barbara: National Center for Geographic Information and Analysis.

Hamilton, G. and M. Worboys, 1996. User Perspectives on Geographic Space. *Proceedings GISRUK'96*, Canterbury: University of Kent.

Kuhn, W. (1992): Paradigms of GIS Use. In: Proceedings 5th International Symposium on Spatial Data Handling. Charleston, S.91-103.

Laurini, R. and D. Thompson, 1992. *Fundamentals of spatial information systems*. London: Academic Press.

De Man, E., 1988. Establishing a geographic information system in relation to its use. In *International Journal of Geographical Information Systems*, **3**, 257.

Mark, D. and M. Gould, 1991. Interacting with Geographic Information: a commentary. *Photogrammetric Engineering & Remote Sensing*, 57:1427-1430.

Medyckyj-Scott, D. and H. Hearnshaw, 1993. *Human Factors in Geographical Information Systems*. London: Belhaven Press.

Monmonnier, M., 1989. Graphic scripts for the sequenced visualization of geographic data. In *Proceedings GIS/LIS'89*, pp. 381-389, Falls Church: ASPRS/ACSM.

OGF, 1993. The Open Geodata Interoperability Specification, Version 1.0, preliminary draft, November 15, 1993. Open GIS Foundation, Cambridge, MA.

Rhind, D., and N. Green, 1988. Design of a geographical information system for a heterogeneous scientific community. In *International Journal of Geographical Information Systems*, **2**, 175.

Schenkelaars, V., 1994. Query classification, a first step towards a graphical interaction language. In Molenaar, M. and S. de Hoop (Eds.) *Advanced Geographic Data Modelling*, pp. 53-65.

Timpf, S. and A. Frank, 1995. A multi-scale DAG for cartographic objects. In *Proceedings ACSM/ASPRS Annual Convention and Exposition Technical Papers*, Vol. 4 (Auto-Carto 12), pp. 157-163.

Tomlin, D., 1990. *Geographic Information Systems and Cartographic Modeling*. Englewood Cliffs: Prentice Hall.

Turk, A., 1992. *GIS Cogency*: cognitive ergonomics in geographic information systems. Unpublished dissertation. Melbourne: Department of Surveying and Land Information, University of Melbourne.

Unwin, D., 1990. A syllabus for teaching geographical information systems. In *International Journal of Geographical Information Systems*, **4**, 461-462.

Voisard, A., 1995. Open GIS: A tool for environmental information management. In Kremers, H. and W. Pillmann (Eds.), *Space and Time in Environmental Information Systems*, 9th International Symposium on Computer Science for Environmental Protection, pp. 160-167.

Yuan, M. and J. Albrecht, 1995, Structuring geographic information and GIS operations. In Frank, A. and W. Kuhn (Eds.) *Spatial Information Theory: A Theoretical Basis for GIS,* pp. 107-122, Berlin: Springer.