INTEGRATED GIS MAINTENANCE MANAGEMENT SYSTEMS

Sponsored by

Utah Department of Transportation
UT-98.02

Mountain Plains Consortium
U.S. Dot University Centers Program

A Utah Transportation Center Project

Performed By:

Computer Based Intelligent Technology Lab.
Utah State University

Energy & Geoscience Institute Geomatics Lab.
University of Utah

February 1998

Disclaimer

TABLE OF CONTENTS

LIST OF FIGURES

INTRODUCTION

A variety of computer tools are being used by UDOT to serve diverse maintenance management needs. Many of the tools are individually tailored for a specific purpose and lack the functionality to share data with other applications. This has resulted in the establishment of numerous databases which are optimized for focused objectives, but unavailable for broader management needs.

Figure 1 was presented to the UDOT Technical Advisory Committee in the fall of 1996. It depicts nine computer task modules (computer tools) being used or having potential use in the Maintenance Division. They are represented by the shapes across the top of the figure. Three types of standard development tools that have potential use for the implementation of broader management analyses are represented by the ellipses at the bottom of the figure. The intent of the figure is to illustrate that a common backbone, or open architecture, would be needed if the data associated with each of the different task modules were to be made available for analysis by standard development tools. The Technical Advisory Committee agreed that the specific objectives of this study would be:

1) To examine the Feature Inventory module in the COBOL-based Maintenance Management System (MMS) and translate the COBOL files into formats that can be used directly by Geographical Information Systems.
2) To develop an ArcView application for displaying the translated feature inventory data on a map of the state routes.
3) To develop a custom task module that directly accesses the translated feature inventory data and displays it on a map of the state routes.
4) To recommend a strategy to transition from the current situation to one in which data from specific task modules is available for broader management purposes.
5) To include the Orange Book and Inhouse Maintenance data into this project; resources became available from another project In the winter of 1997.

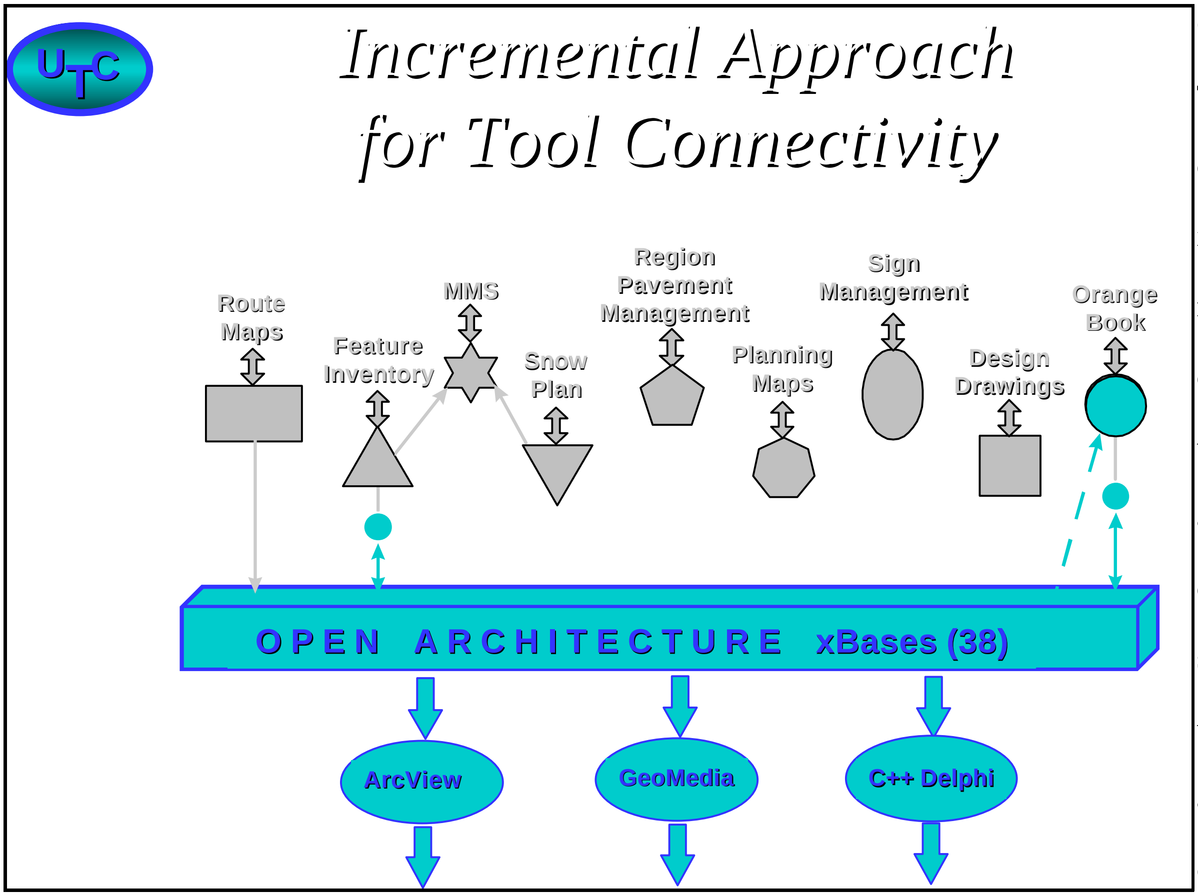


Figure 1. Conceptual Representation of the Problem.

6) To deliver data and software utilities to UDOT.

DATA RESTRUCTURING

Procedure

Figure 2 illustrates the process for restructuring the data files from the Inhouse Maintenance, Features Inventory, Features Description, Orange Book and State Highway Route modules into an open architecture that can be accessed by all standard development tools. The numbers in brackets are keyed to the following discussion.

In-house Maintenance History are stored in MMS COBOL files [1]. An MMS filter [2] was prepared by UDOT to produce a text file of the data [3] occupying approximately 35 Mb. A custom filter [4] was developed to convert the text file into a conventional ".dbf" file [5] occupying approximately 65 Mb. A custom shape filter [6] was developed to merge the route shape file [31] information with the maintenance history data file [5] to produce the In-house Maintenance History shape file [7] occupying approximately 16 Mb.

Feature Inventory and Feature Descriptions reside in separate MMS COBOL files [8] and [15]. The information in the Features Descriptions file is needed to identify and translate the Features Inventory file. Custom filter [16] was developed to directly access the MMS COBOL file and produce a standard ".dbf" file [17]. A custom filter [18] was developed to produce an optimized file of attribute associations [19]. This file was given a ".gdl" extension and occupies approximately 0.02 Mb. A two-step process uses custom filter [9] to directly access the MMS COBOL file to produce a standard ".dbf" file [10], and a second custom filter [11] to split the large file into 37 smaller feature files [12]. The filters require the attribute associations stored in the ".gdl" file [19]. A custom shape file generator [13] was developed to merge the route shape file [31] information with the feature files [12] to produce Feature Inventory Shape Files [14].

Orange Book data were stored in two forms. The 1992-1996 data were stored in conventional Paradox database files. Data for 1986 to 1991 were stored in WordPerfect tables [22]. The Paradox database stored location data in a memo field; consequently, a custom filter [21] was needed to parse the information and convert it to numerical values in the resulting ".dbf" file [24]. A custom filter [23] was developed to work in conjunction with custom filter [21] to merge the 1986-1991 data with the 1992-1996 data into a comprehensive ".dbf" file [24] occupying 0.4 Mb. A custom shape file generator [25] was developed to merge the route shape file [31] information with the Orange Book data to produce the Orange Book Shape File [26].

The State Highway Route data were stored in ArcInfo/ArcView route files [27] created by Chris Glazer in the UDOT Research Division. An ArcView filter [28] produced route shape files [29]. A custom filter [30] was developed to produce the modified shape file [31], which optimizes route segmentation for faster access. The modified route shape file [31] is used to provide geospatial information for custom shape file generators [6], [12] and [25].

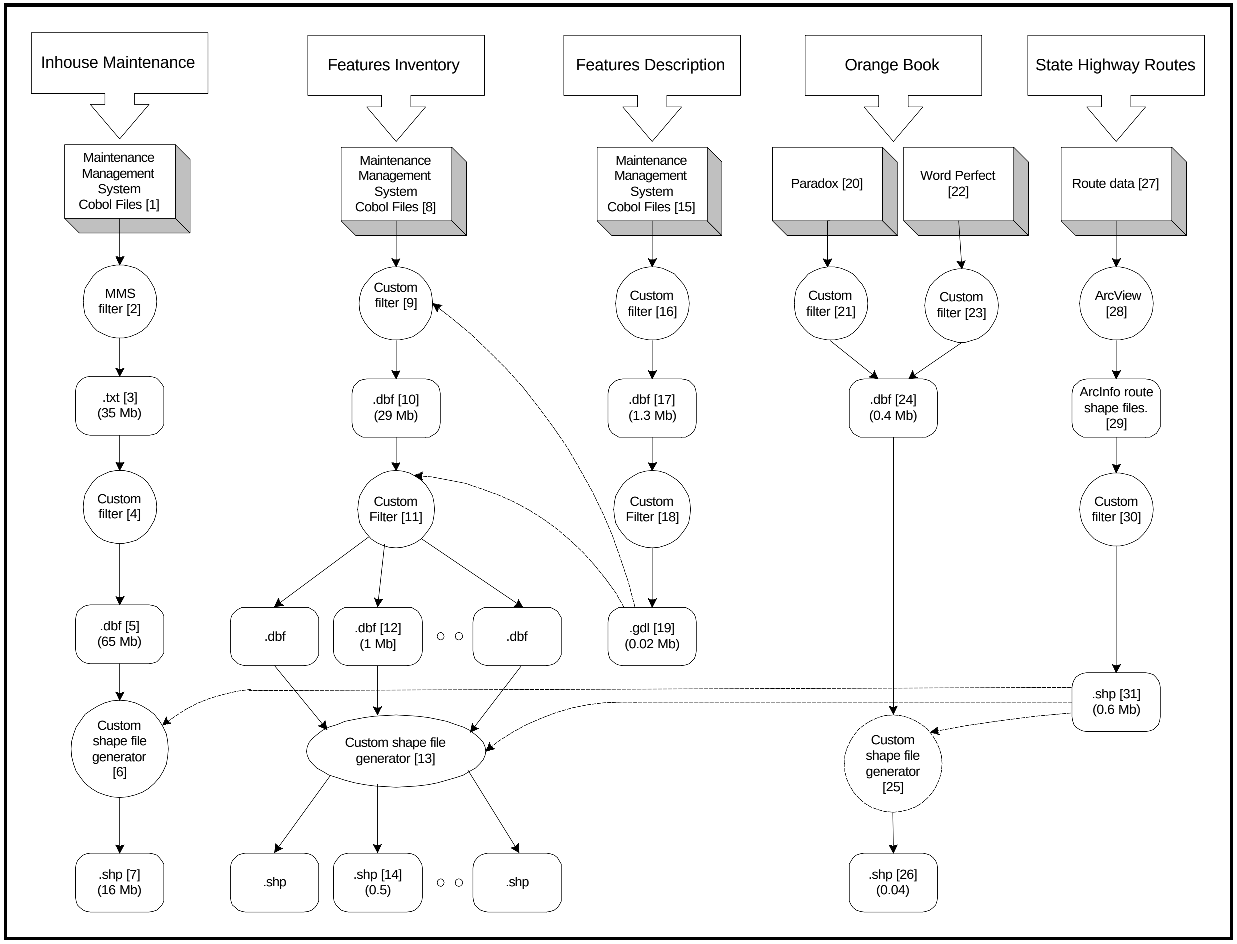
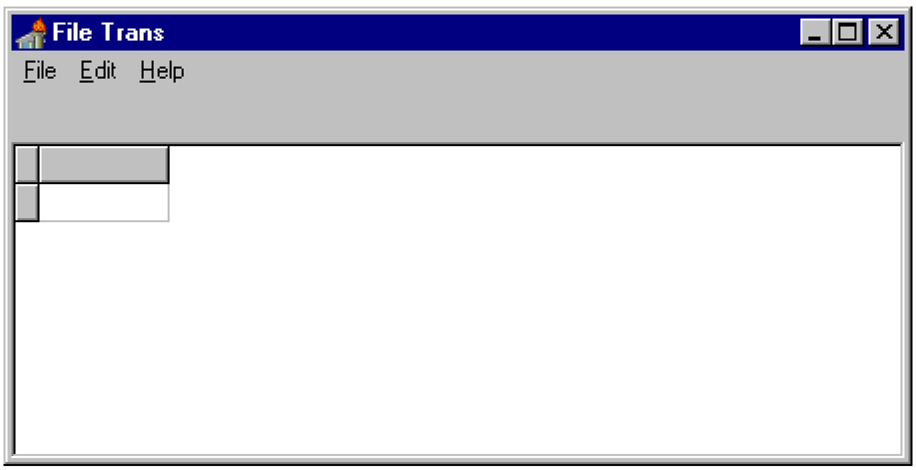


Figure 2. Data Translation Process

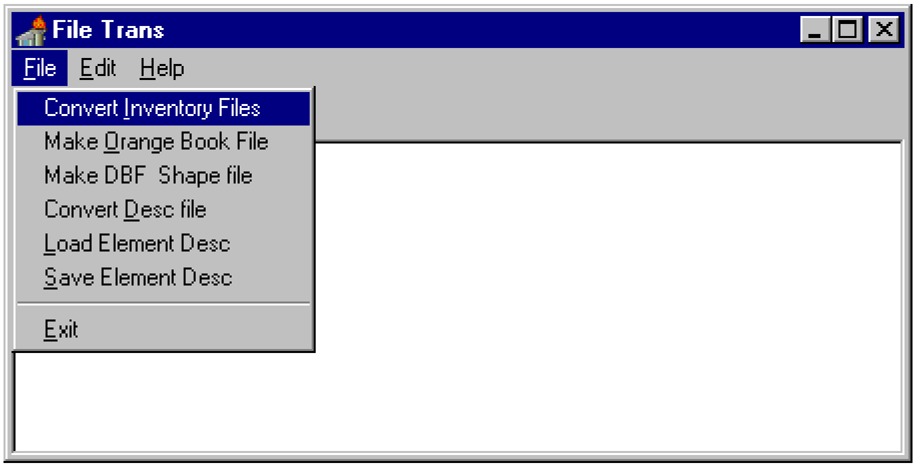
Custom Software Filters

The custom software filters described in the preceding section subsequently were combined into a single program to make future data translations easier. It is called the File Translation Program and converts the data stored in the Reference Inventory and Feature Inventory files into a dbase file with an associated shape file. It also converts the Orange Book paradox file into a dbase file with its associated shape file.

Below are screen shots from the program and explanations of the menu functions. The opening screen is shown here, followed by a display of the expanded "File" menu item.

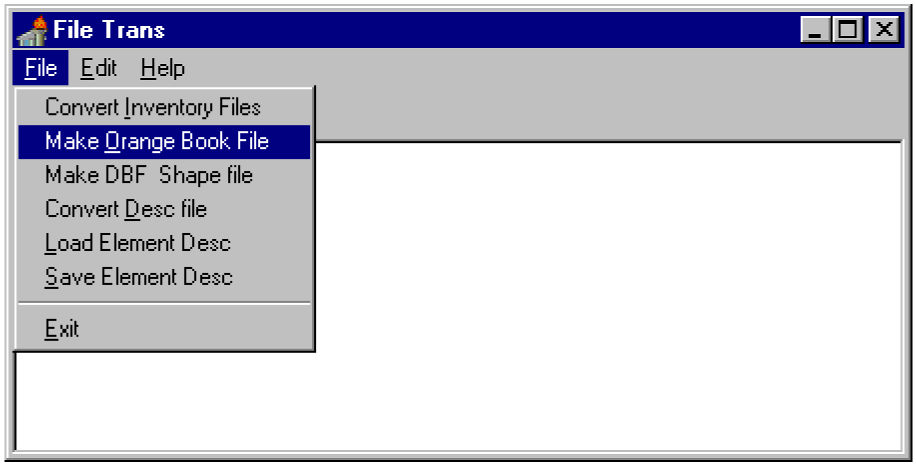


- If a description file is not already selected you are prompted to select one.
- Then you are prompted to select the Reference Inventory File that is stored in a COBOL flat file and is named MFISREFI.dat.
- You are then prompted to select the Feature Inventory File that is another COBOL flat file and is named MFISINVI.dat.



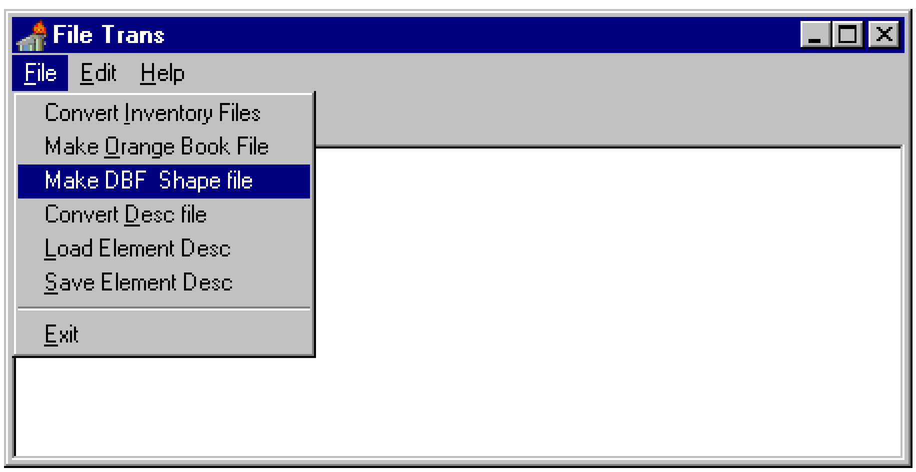
Convert Inventory Files

- If a route shape file is not already selected you are prompted to select one.
- The program then converts the data in MFISINVI.dat to a Dbase file named Master.dbf. After generating Master.dbf the data is broken up into smaller files based on item type. Shape files are then generated for the smaller files.
- The program then converts the data in MFISREFI.dat to a Dbase file named MastRefi.dbf. After generating Master.dbf the data is broken up into smaller files based on item type. Shape files are then generated for the smaller files.



Make Orange Book File

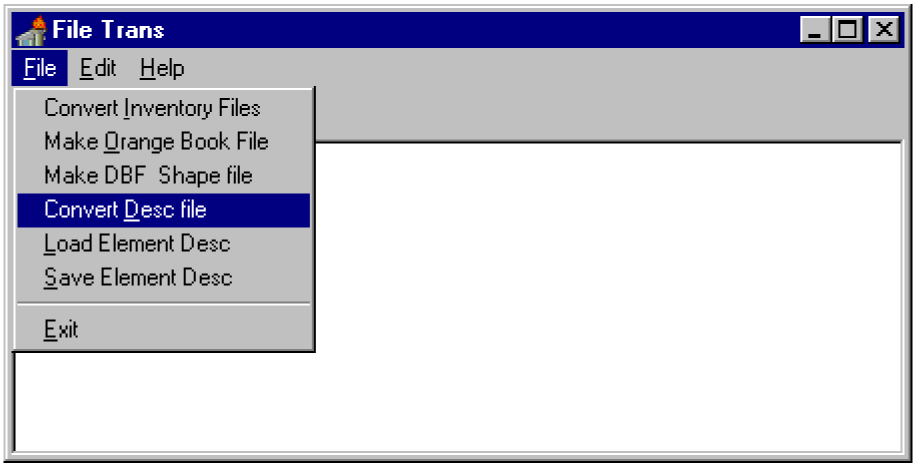
- If a route shape file is not already selected you are prompted to select one.
- Then you are prompted to select the Orange Book file stored in a Paradox file.



- The Paradox file is then converted to a Dbase file from which a Shape file is created.

Make DBF Shape File

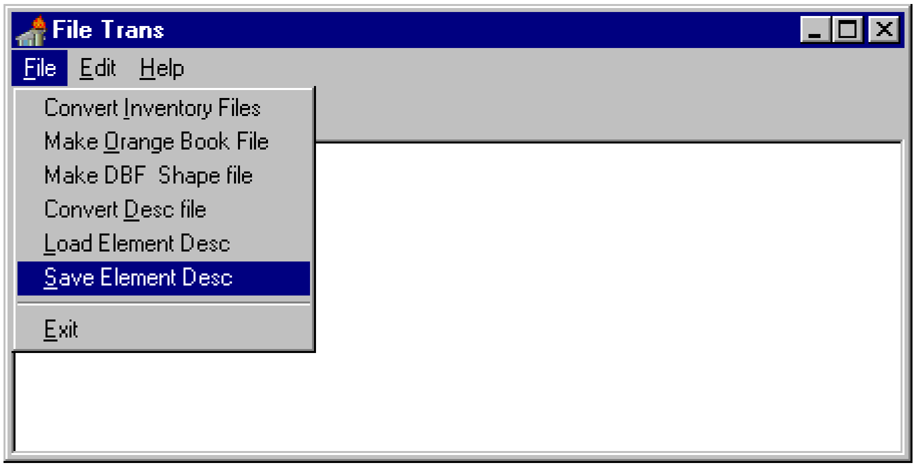
- If a route shape file is not already selected you are prompted to select one.



- Then you are prompted to select the Dbase file.
- A Shape file is then generated from the Dbase file.

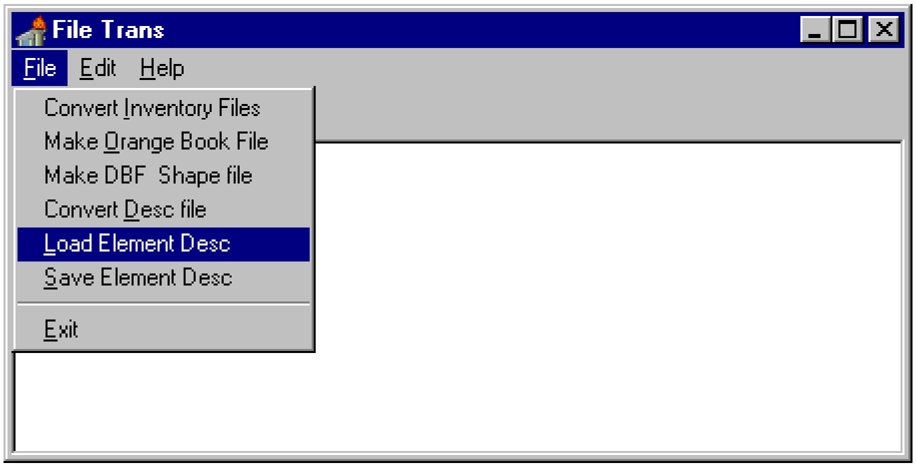
Convert Desc File

- You are prompted for a Description file, which is a COBOL flat file named MFISDESC.dat.
- A new description file is then created with the values in the COBOL file.
- To save this description file use the menus File | Save Element Desc.



Load Element Desc

Prompts for the selection of a Description file and loads it as the active description file.



Save Element Desc

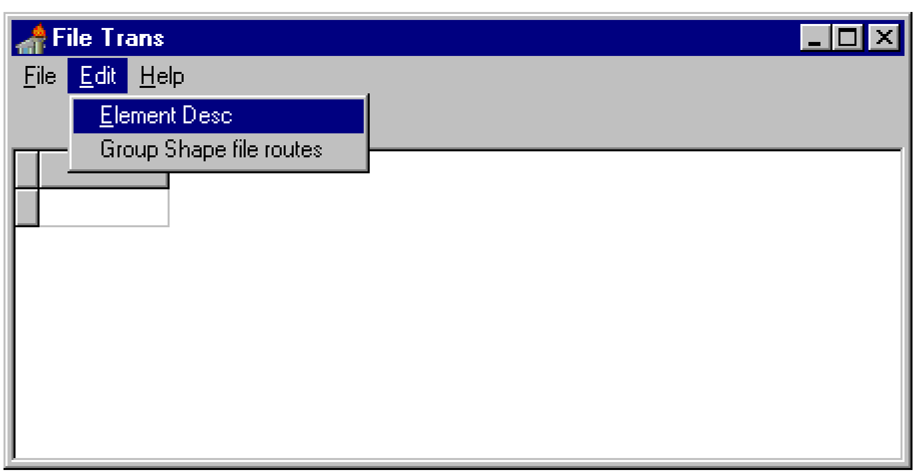
If a description file is in use then you are prompted for the name of the file to save it as.

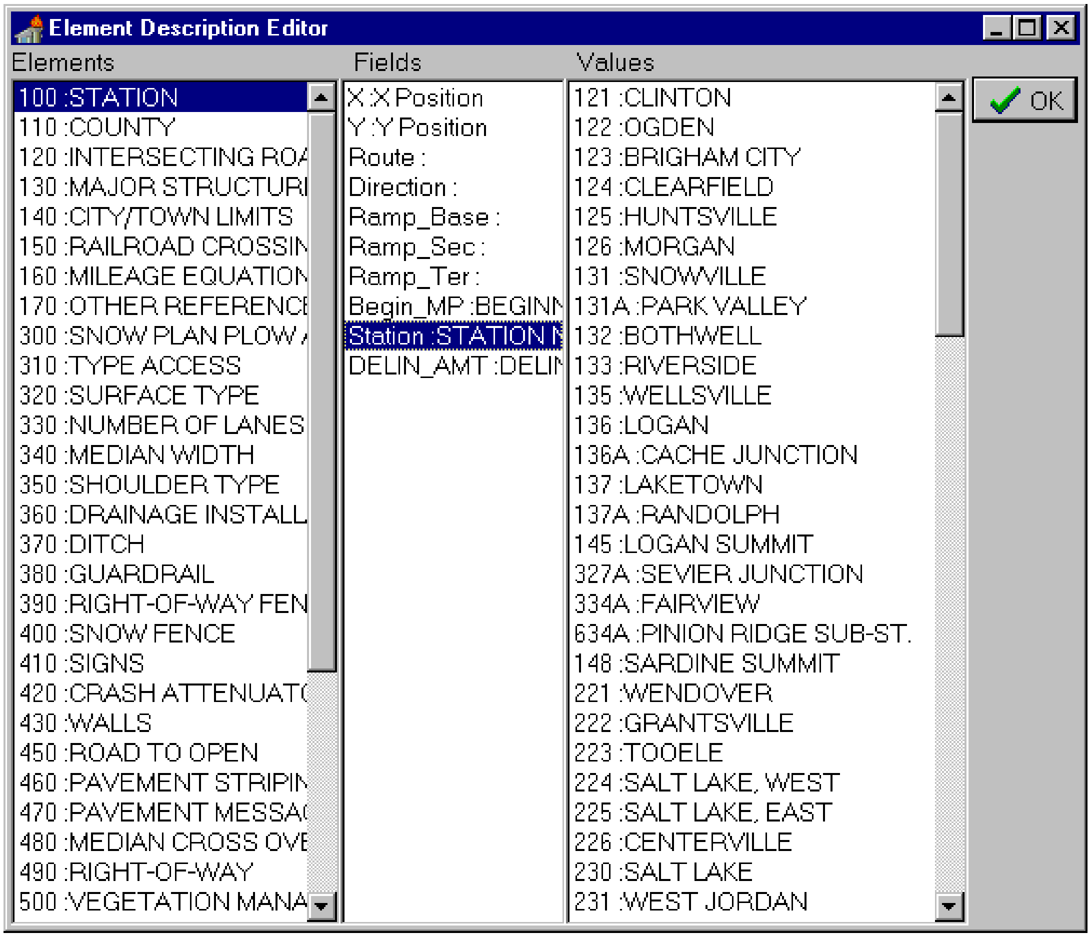
The expanded menu for the Edit menu is shown below.

Element Desc

- If a description file is not already selected you are prompted to select one.
- Then it shows the Element Description Editor

Below is a view of the Description Editor.



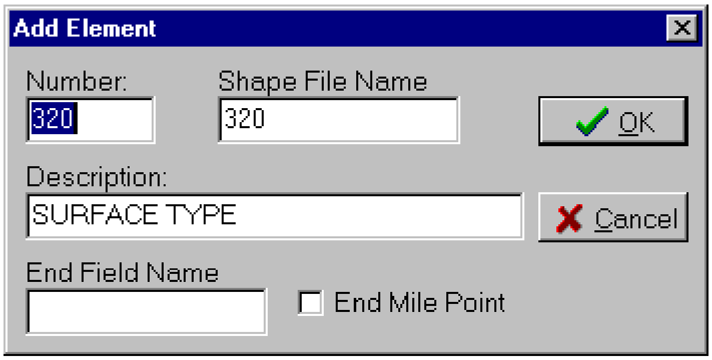


The description editor is made up of three lists. The first list contains all the elements to be viewed, the second contains the fields in the selected element, and the last displays the different values of the selected field.

Right clicking on any of the lists causes a pop-up menu to appear. The menu has three choices: Add, Edit and Delete. Clicking the Add menu item causes the item editor to open for the selected list. Clicking the Edit menu item causes the item editor to open for the active list, and subsequently fill it with the data for the item selected. Clicking the Delete menu item deletes the selected item in the list.

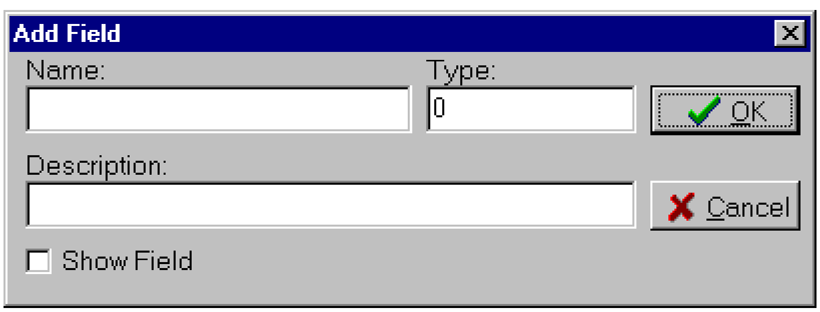
Data File Structures

Modified data file structures are published in Appendices A and B. Following are descriptions of



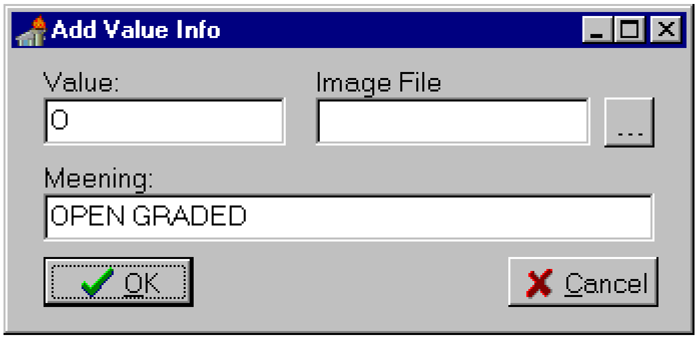
the dialog boxes in File Translation Program.

The Element editor has five fields. The first field Number stores the number used to represent this element in the MMS program. The Shape File Name is the name of the file that holds the data for this element. The Description is the text to show when describing this element in the GIS demo program. The End Field Name is the name of the field that has the length or end mile point for the element. The End Mile



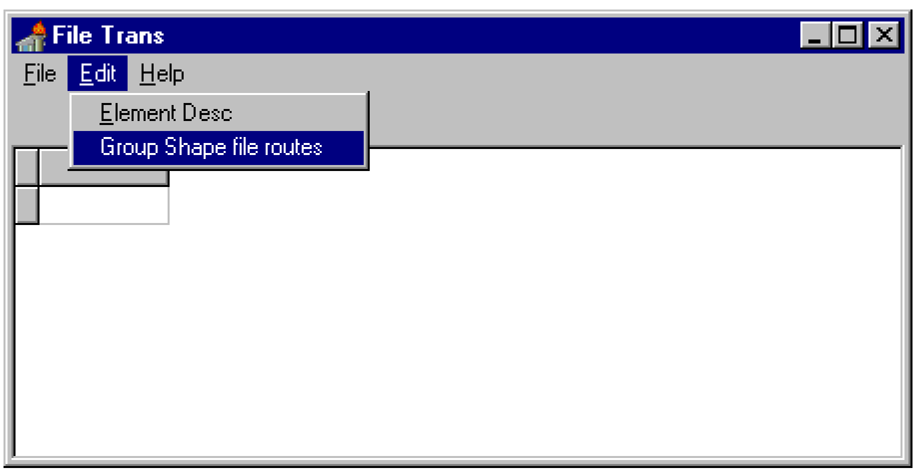
Point is checked if the field referred to by End Field Name contains the end mile point.

The Field editor has four fields. The first field Name stores the name of the field it describes. The Type is descriptor an integer - if it is 0 then the field is an alphanumeric, if it is 1 then the field is an integer number, and if it is 2 then the field is a real number. The Description is the text to show when describing this element in the GIS demo program. The Show Field is checked if the field is to be shown in the GIS



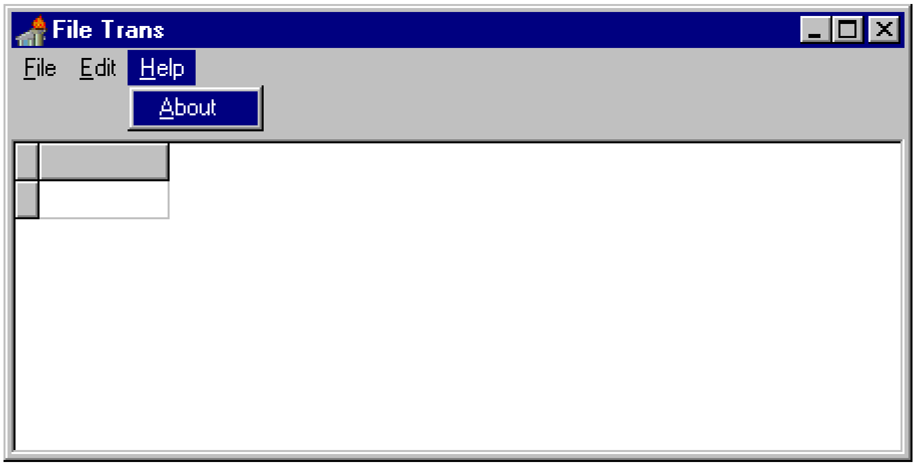
demo program.

The Value Information editor has three fields. The first field Value stores a value the field can have. The Image File is the name of a bit map to show if this value is displayed on the map in the GIS



demo program. If Image File is left blank then it uses the default display. The Meaning is the text to show when describing this element in the GIS demo program.

Group Shape File Routes



It prompts for the route shape file to load, then searches the route file for segments of routes that

have end points in common and combines them.

The help menu item simply contain the “About Box” for this program.

Results

The software utilities developed during this part of the study efficiently translate the special purpose databases into databases with common ".dbf" formats. The translation process accurately reproduces each data field and its contents from the original to the new in essentially a one-to-one relationship. Additional software subsequently processes the databases with the State Highway Route Shape file to produce a suite of attribute shape files.

Once the data are available in conventional formats then comprehensive displays of the data could be generated. The displays exposed several deficiencies in the data that inhibited the comprehensive analyses and displays:

1) Missing Data: data fields were available, but no data had been entered.
2) Missing Data Fields: data fields were not available for important data items.
3) Redundant Data: the same data was stored separately in more than one place resulting in increased data maintenance and the potential for inconsistencies in the data.
4) Inappropriate Data Formats: data were stored in fields with text formats rather than value formats.
5) Inadequate Location Data: location data were missing or stored in non-standard, conflicting coordinate systems.

DATA ANALYSIS TOOLS: CONVENTIONAL GIS

Introduction

The Energy & Geoscience Institute Geomatics Laboratory, Department of Civil and Environmental Engineering, University of Utah, is partnering with the C-Bit Laboratory, Utah Transportation Center, Utah State University, to develop a GIS interface for specific UDOT needs. ArcView™, a commercial GIS tool marketed by Environmental Systems Research Institute, Inc. (ESRI) was selected for the study. The Geomatics laboratory participation has primarily been in the testing of conventional GIS methods of data display and query, and data analysis culminating in analog map generation. This has included assessment of different methods of data incorporation.

NOTE: Throughout the remainder of this section all terms specific to ArcView will be italicized.

Background

The Utah Department of Transportation (UDOT) currently is involved in a serious effort to incorporate GIS into its operating procedures. GIS will be an effective method of allowing UDOT to (1) standardize and update old geospatial databases, (2) build a new geospatial database within the standardized environment, (3) access, retrieve, and input data into the database via an user-friendly geographically based GUI, (4) rapid task-specific data queries, (5) spatial analysis capabilities, including routing and environmental analysis, and (6) high-quality map generation to name a few.

As GIS technology and applications are not easy to learn, GIS specialists are needed for GIS development implementation. However, the use of relatively user-friendly GUI based GIS software, such as ArcView™, gives even casual users accessibility to complex GIS databases. Accessibility and efficiency of GIS database use can be even further enhanced through the incorporation of task-specific modules generated with script languages like ArcView Avenue™, conventional languages like Visual Basic™, and object-oriented programming languages like Delphi™ and Visual C++.

In developing a useful GIS database, especially when incorporating new task-specific modules, care must be taken in designing the GIS properly to insure maximum efficiency. To this end, the Geomatics Laboratory designed several experiments to help determine the best approach to GIS design aid in developing task-specific modules for use with ArcView.™

In addition to the above evaluation, data sets developed from UDOT's Orange Book and COBOL databases has been analyzed in a GIS environment to produce specific maps. These maps compliment the object oriented based interface developed by the C-Bit laboratory.

GIS Data Incorporation/Format Evaluation

ArcView™ was the chosen GIS software platform for this project. This decision was based on several criteria including (1) ease of use for non GIS professionals, (2) UDOT's choice of ArcInfo™ and ArcView™ as their in-house GIS platforms, (3) data format conformity with the State of Utah's Automated Geographic Reference Center's SGID digital geospatial database, and (4) availability of, and ease of incorporation of ESRI's Map Objects™.

UDOT provided data that were being stored in a proprietary COBOL-based database, relating to features inventory, for inclusion in the new GIS-based system. The COBOL data were extracted and reformated into an xbase formated data structure by the C-Bit Laboratory.

Investigations were initiated regarding the most appropriate method of inclusion of the data in a GIS environment. This included considering ArcView™ *event themes* and ArcView™ native *shape* files (thematic layers). The database data were attached to the state roads' ArcInfo™ route system, prepared by UDOT, as points related by milepost and route number. Methods were judged by complexity and by the time involved in generating the respective GIS data. In considering efficiency one must consider the time required to incorporate new data into the GIS, the time to draw a geographic representation of the data on the screen, the time required to select or unselect data on the screen, and the time to open the ArcView™ GIS “project” (compilation of thematic layers).

ArcView™ Event Themes

ArcView™ can incorporate raw database data in the GIS environment as event *themes*. To accomplish this the data must have two fields with xy data, such as longitude and latitude, or in this case milepost and route number, which can be attached to a *view* made up of a *theme* or *themes* consisting of points, lines, or polygons that have already been prepared using traditional GIS techniques. For relatively small data sets this is an effective method of database usage in a GIS environment. However, large data sets are not efficient to incorporate this method. For instance, the following times were recorded using *event themes* derived from the data extracted from the UDOT Cobol-based database. Testing was accomplished using a Pentium 66 computer with 24 megabyts of RAM. *Event themes* were posted using the full state route system. Test A was done using a *project* with one *event theme*. Test B was done using a *project* containing two *event themes*.

Test A_1. Opening an ArcView™ *project* using *event theme* 460.dbf (3,869,345 bytes)

1. Spatial index building - 3 minutes, 44 seconds
2. Drawing lines (roads) and points (feature inventory item) - 3 minutes, 1 second

Total opening time - 6 minutes, 45 seconds

Test $A_{2.}$ Select and unselect all points

1. Select points - 5 minutes, 20 seconds
2. Unselect points - 2 minutes, 45 seconds

Test B_1. Opening an ArcView™ *project* using *event themes* 460.dbf (3,869,345 bytes) and 490.dbf (1,021,928 bytes)

1. Spatial index building (460.dbf) - 3 minutes, 3 seconds
2. Spatial index building (490.dbf) - 1 minute, 17 seconds
3. Drawing lines and points on screen - 4 minutes, 10 seconds

Total opening time - 8 minutes, 30 seconds

Test B_2 . Select and unselect all points

1. Select points (460.dbf) - 7 minutes, 4 seconds
2. Select points (490.dbf) - 2 minutes, 10 seconds
3. Unselect points (460.dbf) - 4 minutes, 42 second
4. Unselect points (490.dbf) - 1 minute, 7 seconds

This method of ArcView™ *project* development is not efficient for day-to-day use and access of the GIS database. As the UDOT GIS database generated by this project could be more than 100 megabytes when finished, the times to access the data as *event themes* would not be acceptable. Therefore the efficiency of using native ArcView™ *shape* files was scrutinized.

ArcView™ Shape Files

Tests similar to those conducted to analyze the use of *event themes* were again used on native ArcView™ *shape* files. The results are reported below.

Test A_1. Opening an ArcView™ *project* using *event theme* 460.dbf (3,869,345 bytes)

1. Total opening time - 34 seconds (no spatial index building necessary)

Test $A_{2.}$ Select and unselect all points

1. Select points - 28 seconds

2. Unselect points - 29 seconds

Test B_1. Opening an ArcView™ *project* using *event themes* 460.dbf (3,869,345 bytes) and 490.dbf (1,021,928 bytes)

1. Total opening time - 42 seconds (no spatial index building needed)

Test B_2 . Select and unselect all points

1. Select points (460.dbf) - 44 seconds

2. Select points (490.dbf) - 17 seconds

3. Unselect points (460.dbf) - 40 seconds

4. Unselect points (490.dbf) - 15 seconds

It is obvious that the native ArcView™ *shape* file format is much more efficient than the *event theme* format. However, the *shape* files were created by selecting all points in an *event theme* and converting them to a *shape* file. This takes a few minutes, but only needs to be done upon updating after the initial file is created.

Analytic Use of GIS/Map Production

Map Generation

Data from dBase files extracted from the COBOl and Orange Book were analyzed for certain features and maintenance items. The following maps were produced from the analyses:

1. Road surface types

2. Road striping

3. Rejuvenation

4. Slurry seal

5. Microsurface

6. Overlay

7. PMSC (plant mix seal coat)

8. Chip seal

9. State routes w/color image backdrop

The maps are color-coded by type or by year as indicated in individual map legends. All maps were generated at a scale of 1:550,000. The maps are suitable for general purposes and are best used in conjunction with the GIS interface for maximum accuracy. It is suggested that the maps be used to quickly assess the status of the feature or maintenance item followed by a GIS query for verification and to access ancillary information. The paper maps allow the user to quickly relate map items to the computer screen. The maps give an overall idea of the status of the various mapped items at a regional or subregional scale, but lack minute details. This is due to the generalization that is inherent to the map scale. All paper maps are generalized to varying degrees. However, the digital GIS is capable of containing details, spatial and thematic, at a scale of 1:1. This is not always practical in reality, but the GIS will nearly always carry more detail than paper maps produced from it.

Problems

The map production phase of this project illuminated problems in the data derived from the Orange Book and COBOL databases. The problems arise as the databases originally were not created with the generation of graphical geospatial data in mind. Specific problems included lack of standardization in field entries, data redundancy, and the incorporation of disparate, albeit related, data in individual data sets. To fully and accurately use the data in a GIS environment, these problems must be addressed. The problems frustrate efficient data analysis and slow map production significantly. Suggestions will be offered in the Conclusions and Suggestions section below. The above problems do not preclude the use of the data sets, as is, for immediate query, but use should be tempered by the knowledge that inconsistencies exist.

Results

ArcView™ was applied to the translated data files. Numerous data discrepancies were encountered primarily because the original databases originally were not created with the generation of graphical geospatial data in mind. For example, data sets produced from the Orange Book and COBOL files could be improved greatly by:

1) The standardization of descriptive terms per field, i.e. either FY1996 or 1996, but not both.
2) The deletion of redundant records.
3) The generation of new data set by splitting out disparate data, even if closely related.
4) The development of intuitive descriptors and field headings.

The efficiency of the GIS database created in this project is maximized by use of *shape* files. Efficiency also could be increased through the use of task-specific modules operating outside of ArcView™ made accessible via the ArcView™ menu. The following methodology for development, maintenance, and updating data would greatly enhance the productivity of ArcView™ applications:

1) Arc/Info® and ArcView® be officially adopted as UDOT's GIS software as it already resides in-house, it is already in use, and its format is a standard in Utah State Government.
2) User specific needs be addressed and incorporated through the ArcView® interface through custom programming and scripting.
3) A specific high-end computer be used as a server to the GIS database.

4) All initial GIS data be stored as *shape* files with associated xbase files, or other openly accessible standard format.

5) An automated procedure would be initiated to create *event themes* upon entry of new data in the xbase files and to recreate and overwrite the old *shape* files.

6) End-users each be provided with a specific ArcView™ *project* with their specific task module(s) included.

7) GIS integration into a UDOT central database (Informix®) with access to Microstation® CAD drawings through the use of ESRI SDE® or equivalent be promptly investigated.

DATA ANALYSIS TOOLS: CUSTOM MODULE WITH MAP OBJECTS

A review of the trade literature indicated that the ESRI Map Objects Software would be the most promising Application Programmers Interface (API) for customizing stand-alone applications based on the ArcInfo route files. Map Objects was evaluated by developing menus and hot buttons to perform functions that otherwise would take a working knowledge of a commercial GIS tool like ArcView. The application was designed specifically for displaying information from features inventory, orange book, and inhouse maintenance on maps generated from the ArcInfo route files.

This is a prototype developed for the purpose of demonstrating and evaluating new technologies, and is not a final operational system. The application is described in this section and the software is provided on the CDROM accompanying this report.

The User Interface

Figure 3 is the view of the main screen. A list of features, including those from the Orange Book and In-house Maintenance files, is displayed on the left. "Walls" has been highlighted and three predefined classifications in "Walls" are displayed at the bottom of the main list: "TYPE WALL," "AVERAGE HEIGHT," and "TYPE MATERIAL" "TYPE MATERIAL" has been highlighted and fields in this classification are displayed "CONCRETE", "WOOD", AND "OTHER." A map of the state route file is displayed on the right. Although items have been highlighted in the features list, they have not been selected and so the location of the features are not yet displayed on the map.

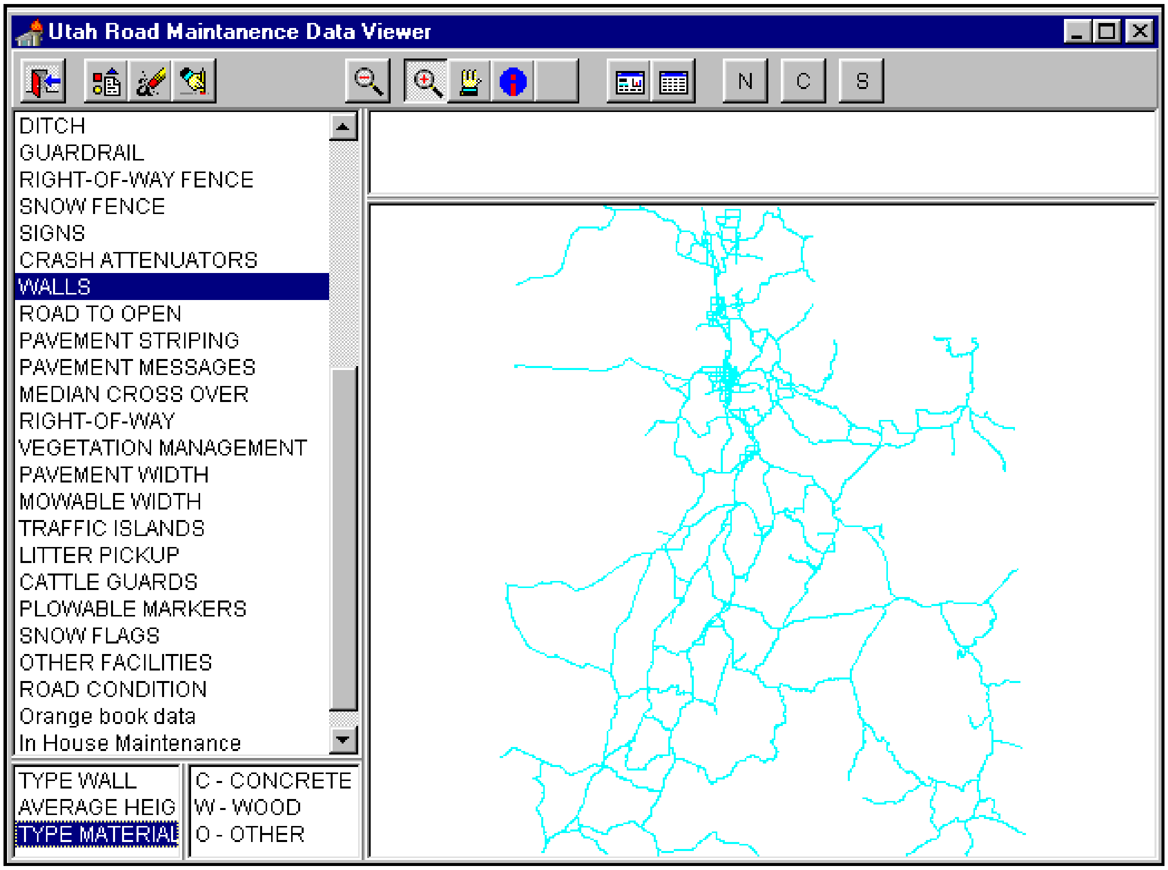
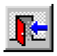


Figure 3. Display of the Main View

A number of "hot buttons" are available to the user at the top of the window:

 Exit the program.

 Display the selected features onto the route map.

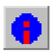
 Clear the display of all features from the route map.

 Clear a selected feature from the route map.

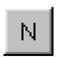
 Zoom out.

 Zoom in on a selected area.

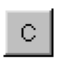
 Move the map in the view window.

 View the data fields associated with a displayed item.

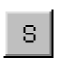
 Display a bitmap image behind the route map.



Nullify all selections.



Clear all feature displays.



Search for data in selected items using SQL commands.

Figure 4 shows the view after a bitmap of a typical Utah road map has been inserted behind the route map and the "Wall" features have been displayed. The road map is low resolution, but demonstrates the functionality. Much higher resolution maps could be inserted just as easily. The view shows the clustering of walls in the Salt Lake City area.

Figure 5 shows the view of an enlarged display of the Salt Lake City area. This was achieved using the "zoom in" hot button. Also shown is the "Search" window. The Search function allows the user to enter standard SQL commands to select and display data. Figure 5 demonstrates the SQL command to select all walls having a material type of concrete. "TYPE2" is the name of the field and "C" is the field value indicating concrete. The field name and the symbol denoting concrete were established in the original features inventory files.

Figure 6 shows the view after the search for all concrete walls has been completed. A table is displayed containing all the data fields associated with all concrete walls. The names of the fields are displayed at the top of the table columns and the values are shown in the rows. Searches can be performed on multiple fields. The “Select” button on the right of the “Search” window provides the user with means to highlight items in the table and see them highlighted on the map. The “Find” button will cause selected items to blink on the map.

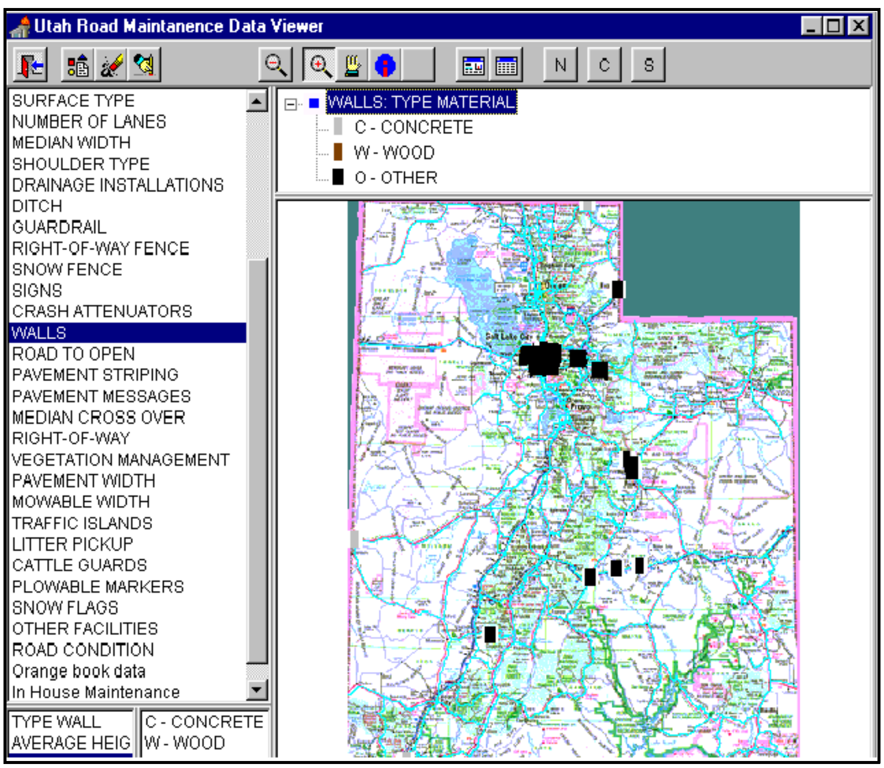


Figure 4. View of the Statewide Display of the “Wall” Features.

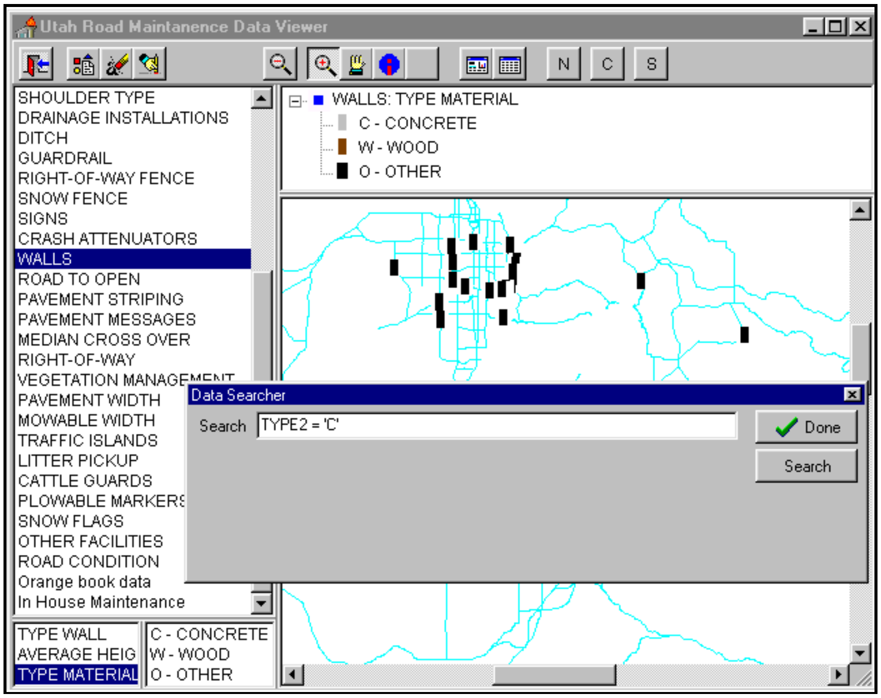


Figure 5. View of an Enlarged Display of the Salt Lake City Area and the Search Window.

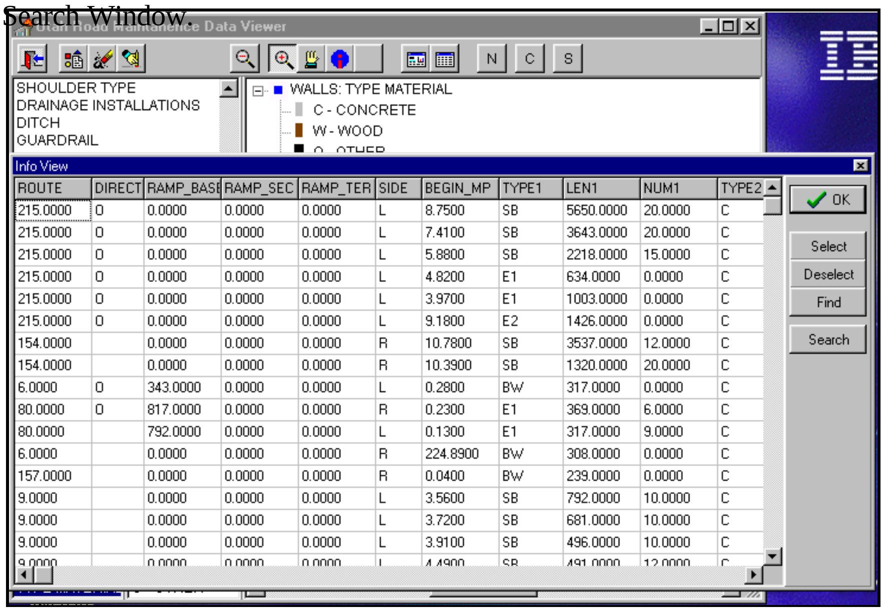


Figure 6. View of the Search Function Results.

Results

Custom programming using ESRI's Map Objects library has strengths and weaknesses. Strengths include the ability for custom programmers to directly access powerful GIS functions without being limited by an awkward interface or constraining script language. More specifically:

1. A compiled, stand-alone executable application performs fast and efficiently at run time.
2. A standard development language (i.e. C++, Delphi. Visual Basic) has much more flexibility and power for tailored numerical and search algorithms.
3. The developer has complete control over the look and feel of the user interface.
4. The GIS functions themselves perform much faster when accessed directly than when accessed through ArcView.

Weaknesses include missing functionality such as:

1. The Map Objects library does not directly support the ArcInfo/ArcView route file system.
2. Although the products are easy to use, the development using Map Objects requires considerable skill using standard programming languages.

RESULTS

A variety of computer tools are being used by the Maintenance Division to serve diverse maintenance management needs. Many of the tools are individually-tailored for a specific purpose and lack the functionality to share data with other applications. This has resulted in the establishment of numerous databases that are optimized for focused objectives, but are unavailable for more comprehensive management needs.

During this study the contents of four separate and unique databases were extracted and translated into a common database format. This format provides accessibility for all conventional analysis and developer tools such as ArcView, Delphi/C++, Visual Basic, and standard database engines. Once the data were available in conventional formats, then comprehensive displays of the data could be generated. These displays exposed several deficiencies in the data: 1) missing data, 2) missing data fields, 3) redundant data, 4) inappropriate data formats, and inadequate location data.. Thus, one of the benefits of visual displays is the exposure of data discrepancies and inconsistencies. The process of accessing and displaying feature attributes on the UDOT State Highway Route map was found to be an effective means for identifying data discrepancies.

Two approaches to analysis were conducted and compared: conventional ArcView and Custom Modules using Map Objects™. ArcView is a powerful analysis tool that can be used for general analyses, as well as for repetitive tasks. ArcView requires a moderate amount of training for general use. It may be customized for repetitive tasks by means of its script language, Avenue™. Custom modules using Map Objects are best suited to repetitive tasks, to applications that rely heavily on numerical algorithms, and to applications that link to other applications by means of system APIs. Loading and execution times are generally faster for custom modules. Custom Modules are easy to use because they normally are developed for a specific task.

Many of the analyses and reports needed by management do not require the functionality of a GIS. When the results of an analysis are best communicated by tables, diagrams, and charts; then the best approach may be a conventional database tool.

When data are “warehoused” in conventional formats, then data collection, data storage, and data analysis/display can be thought of as three more or less independent processes. This strategy is developed further in the next section of this report.

IMPLEMENTATION PLAN

Premise and Strategy

Different levels in the UDOT hierarchy have different needs for data accessibility and therefore, different perspectives for database management systems in the agency. For example, the mission of upper management includes interfacing with the political system, high level allocation of resources, strategic planning, operation of centralized functions such as accounting, and other purposes requiring access to broad-based information to fulfill comprehensive agency requirements. Middle management interfaces with upper management, anticipates trends from historical data, obtains and allocates resources to the operating units within their section, and accomplishes comprehensive reporting duties for their section. The operating units in a middle management section are producing the actual work that is being judged by the public, and its needs are focused on performing specific tasks, frequently under trying conditions.

There is a natural tendency for upper management to desire a large relational database system providing direct access to agency-wide information. At the other end of the spectrum, individuals in operating units desire focused computer tools that are optimized to be fast, efficient and user-friendly for their specific task. This incongruity between centralized accessibility vs. distributed efficiency is exacerbated by the huge number and diversity of tasks performed throughout UDOT. Over time, attempts have been made to implement larger, more comprehensive single-vendor systems. Experience has indicated that the systems often lack specific functionality and are awkward to use by the operating units. On the other hand, individuals in the operating units have been innovative in obtaining and developing software tools that satisfy individual needs; however, without much regard to higher level needs for comprehensive data availability. Software vendors sometimes contribute to the problem by claiming that their data is openly accessible, which may be true in theory, but may be quite limited in practice.

The authors' overriding premise is that it is a practical impossibility to obtain a gigantic turn-key system that satisfies everyone; much less to deploy one if it did exist. Rather, attention should be focused on establishing a skeletal agency-wide architecture that not only permits, but encourages, multi-vendor competition for the operating level, and at the same time assures reasonable data accessibility to upper management. This will be referred to as the “System Integration Architecture.” Implementation of the elements in the architecture would be an evolutionary process so performance at the operating unit level would not be impeded while accessibility at higher levels is gradually enhanced.

Geographic Information Systems (GIS) have become increasingly useful as a means for accessing data from diverse databases. This has transpired because the fundamental purpose of GIS is to display overlays of diverse data on a map, and consequently the GIS vendors have been obliged to develop “filters” for importing data from a variety of popular commercial databases. Although GIS systems are extremely powerful tools for their intended purposes, there are a variety of other purposes, many of which are important for management, that are cumbersome or impossible to accomplish using standard GIS functionality. GIS should be thought of as an extremely effective tool for accessing, analyzing, and displaying data that is tightly coupled with map displays; but not necessarily as the core of a solution to UDOT’s data accessibility needs.

A reasonable approach for a System Integration Architecture (ISA) would rely on the following principles:

1. Operating units would be encouraged to purchase or develop software optimized for their specific task at their level, provided that consideration was given to standardize that task through UDOT regions. A specific task software application is referred to as a “Task Module.”
2. A competitive atmosphere would be maintained for vendors to bid on new and enhanced Task Modules.
3. An agency-wide “Data Warehousing” system would be maintained to interface data in each Operational Task Module with the accessibility needs of higher levels in the UDOT

management hierarchy. The Data Warehouse would be updated periodically by each Operational Task Module at a predefined frequency.

4. The ISA would be designed to accommodate data from the Task Modules so long as the data conformed to a published set of minimum database compatibility requirements. Each Task Module would be required to contain “filters” that would export data to the Data Warehouse in the prescribed format. The operating units would be responsible for getting their data into the Data Warehouse.

5. Higher management levels would develop management Task Modules to access data in the Data Warehouse for their own specific aggregation, analysis, and display needs. A competitive atmosphere would be maintained for vendors to bid on new and enhanced modules at these levels.

Implementation Plan for UDOT Maintenance

Based on findings from this study, the following implementation plan is proposed for the initiation of a System Integration Architecture by the UDOT Maintenance Division.

The objective can be thought of as having two more-or-less distinct requirements: operating level task accomplishment and data export to the Data Warehouse for subsequent use by higher levels in management.

Operating Task Accomplishment

The requirement here is to accomplish specific task objectives at the operating level. This includes data input, efficient data storage, and efficient data analysis and display. This may be accomplished by one or more Operational Task Modules.

Data Collection and Input

1. Define an objective and specify the input data elements needed for this objective. The elements can be text, numerical values, vector graphics, pictures, sound clips, video clips, or any other type of electronic information. Specify a user interface designed to effectively input the elements from either the keyboard or an electronic file.
2. Specify the location data needed for the objective, route-mile post and latitude and longitude coordinates.
3. Specify field data collection requirements such that all information is obtained for the input elements. Where possible field data should be collected in electronic format that can be automatically entered into the module (e.g. bypass the keyboard whenever possible). Separate data collection task modules may be required.

Efficient Data Storage

1. This is a localized database optimized specifically for this objective. Specify that this database will be located on a local computer with connectivity to headquarters. Connectivity mchanisms will be described in detail by the contractor or in-house developer and approved by UDOT ISS prior to development.
2. Specify that the software contractor or in-house developer must describe in detail the file formats they plan to use for efficient data storage with this application. Sufficient detail must be provided so that other developers can access the same database to add and enhance modules. The formats must be approved by UDOT ISS prior to development. Complete descriptions of all data fields must be published by the contractor or in-house developer.

Data Analysis and Display

1. Specify the screen displays to be provided for this objective. Separate analysis/display task modules may be specified.
2. Specify the reports to be generated for this objective. Separate report task modules may be specified.
3. Specify graphical and map output displays.

Export Data to the UDOT Data Warehouse

Specify the format and frequency of data to be delivered to the UDOT Data Warehouse. The contractor or in-house developer must provide an export filter to automatically deliver data to the Warehouse as approved by UDOT ISS prior to development.

APPENDIX A

DATA STRUCTURE FORMATS