



# GEOTECHNICAL DATA MANAGEMENT ISSUES FOR TRANSPORTATION AUTHORITIES

S. Caronna  
gINT Software, Windsor, CA, USA

**ABSTRACT:** For transportation authorities, efficient management of geotechnical data plays a crucial role in an agency's ability to design, build, and maintain facilities in a timely, accurate, and cost-effective manner. Most transportation authorities already use some kind of data management system, and many have encountered varying degrees of problems in the design, usage and maintenance of these systems. This paper extracts some of the principles of database design that have proven successful in maximizing the usability and maintainability of subsurface data management systems.

The paper outlines the common goals of a well-constructed geotechnical data management system for transportation authorities. The paper surveys the primary data management categories including design, collection, manipulation, querying, reporting, interaction with other systems, and dissemination, all of which need to be addressed in a cohesive data management solution. The features of each category are reviewed, and some of the problems that can result from faulty design are delineated.

## 1. INTRODUCTION

Transportation authorities struggle with massive data requirements for information that must be collected, stored, manipulated, and used. Data management tools are available but the quality of these tools is wide ranging. Even given the appropriate tools, implementation quality can vary dramatically. This paper does not recommend any particular data management tool or tools but does address some of the issues that make for an appropriate overall system.

The range of database management practice in transportation authorities today is dramatic. Some are still mainly paper-based. Others have central data repositories with sophisticated querying and reporting tools. Most are somewhere in between with mixes of software and paper-based tools. Significant enhancements to productivity and data integration can be achieved by improving database design, integration, and reporting systems (Hoppe 2003, Plant et al. 1998).

Some of the concepts outlined in this paper may appear obvious and trivial but each has been included because these simple rules are routinely ignored to the detriment of data usability.

## 2. DATA MANAGEMENT GOALS

The end goals of any data management system are to produce useful, accurate, and maintainable information. The first steps in a design of such a system are to address these issues within the context of the needs of the organization.

### 2.1.1 Data Usefulness

How an organization defines what is "useful" depends on needs of the eventual "data consumer". Ideally, each consumer group should be surveyed to discover their needs before a data management solution is implemented, or otherwise the organization risks a solution that is ineffective and unusable. Governmentally mandated requirements that impact the type of data that are stored must also be taken into consideration at early stages.

The degree of usefulness is not just dependent on the type and amount of data but also the querying and reporting tools. Many database systems are lacking either the necessary tools or cannot communicate

with external tools to perform certain tasks such as calculations, statistics, modelling and visualization. For example, it is common practice to extract relevant portions of the data to an external program such as Excel<sup>®</sup> for manual manipulation. This can be a problem because the process is labor intensive and is outside of the QA/QC processes of the system. A better practice is to build in the tools to perform these tasks or set up communications directly from the database system to tools that can perform them. Identifying the processes that require external handling and supplying front end programs that encapsulate these processes directly from the original data eliminates these cumbersome and inefficient tasks.

## 2.1.2 Data Accuracy

To help ensure data accuracy, a number of systems need to be in place. The design of the database must incorporate business rules that reduce the possibility of the insertion of incomplete and illogical data. Obviously no system can ensure that all data are completely accurate. For example, there are no general internal checks that can catch a data entry error of inputting a depth of 5.4 that should have been 4.5. However, many classes of erroneous data can be prevented from being entered and others can be caught before the data are committed to the database. For example, recovery must be greater than or equal to RQD, sands cannot be "stiff" and clays cannot be "dense". Following are some of the available tools.

### 2.1.2.1 Referential Integrity

Setting up appropriate relations between data tables is crucial to avoid, for example, samples not associated with boreholes or test specimens without samples. Projects have boreholes, boreholes have depth-related, time-related, and other data. Proper relationships would ensure that no borehole can exist without an associated project, no sample without an associated borehole, etc. Referential integrity rules can be established within the database front end software program but it is much better to assign this task to the underlying database. There exist, for example, commercial programs using ACCESS<sup>®</sup> for their database that have no relations defined in the ACCESS<sup>®</sup> files. Only the software front end enforces the integrity rules. This is quite dangerous in that other programs manipulating the database directly, or working in ACCESS<sup>®</sup> itself, could create "orphan" records, e.g., samples assigned to borehole "B-123" which is not defined in the database.

### 2.1.2.2 Appropriate Data Types

In determining what is an "appropriate" data type, the first considerations must be to avoid data formats that make accuracy checks and querying impossible. It is common to overload a field by allowing any information to be input. For example, take a field for water depth. This should be a numeric field. Many organizations will make this a text field so that comments like "Not Encountered" can be inserted. This removes a valuable data accuracy tool. The user could type "10.1" ("O" instead of "zero") or "10.1 feet". Both these entries would prevent queries and plotting since they are not numbers. Instead, it is better to make the field numeric and to have another field for water level comments.

With date fields, using a general text field can invalidate the use of date data for any kind of querying or plotting process. This is commonly done so that users can type "3-4 March 2005" or other types of range dates. Text fields also have no built-in date check as a date field would so that erroneous dates such as "30 February 2005" can be inserted. It is better to make date fields date types. If a range is needed, create two date fields.

Another problem with date data in text fields is localization. For example, a date is typed in Toronto as "3/6/2005" for 3 June. The database is opened on a system in San Francisco and the data reads the same but the user assumes this is March 6. With dates stored in date fields, the system automatically adjusts the data to reflect the locale. Therefore, if the same date was entered in a date field in Toronto, the user in San Francisco would see "6/3/2005".

Latitude, Longitude, Station, and Offset are other common cases of the use of text fields for data that would be better stored in numeric fields.

Latitude	Longitude	Station	Offset
44°38'48.23"	79°27'18.65"	15+123.4	8.6 Lt

Figure 1. Coordinates as text.

In Figure 1, using a text format for this data presents numerous problems. First, the data is open to data entry errors. Second, the numeric text string is difficult to query. Third, some effort is required to write validation rules for text strings like this. In the case of Latitude and Longitude, it is also particularly difficult and time-consuming to enter the data because of the need for non-numeric characters.

A better alternative would be to break up the coordinates into components:

Latitude Degrees	Latitude Minutes	Latitude Seconds	Longitude Degrees	Longitude Minutes	Longitude Seconds	Station Major	Station Minor	Offset	Offset Direction
44	38	48.23	79	27	18.65	15	123.4	8.6	Lt

Figure 2. Coordinates as components.

This format can be easily validated and has the additional benefit of helping to eliminate common typographical errors because the fields are just numeric field types and require only that the numbers be input. The "Offset Direction" could also be associated with a lookup list with appropriate choices, so that all the user would have to do is choose one item from a dropdown list to get the appropriate value.

A third approach is to use just decimal numbers:

Latitude	Longitude	Station	Offset
44.646731	79.455181	15123.4	-8.6

Figure 3. Coordinates as decimal numbers.

This format combines the advantages of the component approach (ease of validation and reduction of typographical errors) with a much easier querying ability. A good reporting tool can easily handle the increased complexity of presenting the data.

### 2.1.2.3 Required Fields

It is obviously important that all required data are input. In any modern database product it is simple to tag fields as required. There are also fields that have dependencies that cannot be tagged as required. For example, there may be a field for the RQD value. In this case the field needs to be required only if the sample was a rock core. This type of requirement needs to be enforced in the validation rules (see 2.1.2.8).

### 2.1.2.4 Calculated Fields

Most organizations store recovery and RQD as percentages in their databases. On the reports that is how they will generally be shown. However, that is not how they are collected. The values are derived from length data. The original recovery and RQD are length values and it is best to store these data, not the percentages, in the database. Let the software do the percentage calculation. This approach eliminates a manual calculation step -- which is just one more opportunity for a mistake.

### 2.1.2.5 Valid Value Lists

Whenever possible, it is important to associate valid value lists to fields because they ensure data consistency and enable accurate querying. These "lookup" tables force the user to select from the list. This reduces the chances of incorrect data entry and enforces consistency. Assume there is a "Drilling Method" field in the database and for a particular borehole the method is "Hollow Stem Auger." If left on their own, users may type "Hollow Stem Auger," "Hollow Stem," "HAS," "H.S.A.," or "Halow Stam Augir." An organization would decide the official spelling and that is the only choice the user would have. The consistency quality of valid value lists also makes querying possible. A query asking for all holes drilled with a hollow stem auger cannot be written without knowing how the entry is spelled and not knowing if everyone spelled it the same way.

### 2.1.2.6 Ranges on Numeric Data

Where appropriate, assign valid ranges to numeric information. For example, sample recoveries must be between 0 and 100. These checks are generally quite easy to implement and can remove a large class of inappropriate data.

### 2.1.2.7 Attribute Tables

Some data are associated with particular attributes, for example, casing properties associated with each borehole: material, outer and inner diameters, and driving mechanism. All the appropriate fields could be added to the borehole table. Generally, however, these properties are linked, that is, there is a finite set of casings used and for each casing type these values are constant. In this case, it is better just to have one casing field that links to a casing table. A casing identifier would be assigned to each casing type and that is the only data that resides in the main database. Furthermore, the available casing identifiers would be in a valid values list.

Hole ID	Type	Final Depth (m)	Casing ID	Local X (m)	Local Y (m)	Local Z (m)
B-1	DCP	7.5	A-1	653994.6	123248.5	10.89
B-2	RC	9.5	B-27	654200	123300	16.23
B-3	RO	20	A-1	150 153 136	23027.5	9.87
B-4	RO	20	B-27	200 205 68	23087.5	16.52
B-5	RO	20	C-82	150 155 68	123287.5	20.42
B-6	RO	20	B-27	654699.2	123480	26.75
B-7	RO	20	A-1	654701.7	123661.3	30.58
B-8	SCP	14.07	B-27	654511	123512	24.56
MW-1	RC	14.6	C-82	654600	123600	27.34
MW-2	RC	3.8	A-1	654413	123415	20.78

Figure 4. Casing ID lookup during data entry.

The other attributes can then be extracted on reports by looking up the appropriate record in the casing table for the current casing identifier. This approach speeds up input and prevents the user from assigning an improper attribute to a casing. Many opportunities exist for such attribute tables: sample types, clients, and drilling methods, to name a few.

### 2.1.2.8 Validation Rules

Many of the tools outlined above validate data automatically in some manner. For example, one cannot type "abc" in a numeric field and ranges can be set up to prevent out-of-range numeric data from being input. However, there is a deeper level of validation for which specific validation rules are required. For

example, there are many cases of validation that require examining dependencies. A percentage recovery of 50 is a valid value. However, if an RQD for the same sample is entered as 75, one of those two numbers is invalid. A sample depth of 45.5 is valid, but not if the total depth of the hole is 32. A moisture content percentage of 29 and a dry density of 1.59 Mg/m<sup>3</sup> are both valid values but if the specific gravity is 2.65, then the degree of saturation is 115% which invalidates at least one of the three values.

There are many validation dependencies in geotechnical databases, and so it is crucial that such rules be written to obtain accurate data.

### 2.1.3 Maintainability

Databases are not static entities. An organization's needs change with time and the database and the management system must change for it to remain relevant. For an agency to have its data keep pace with changing needs, the system needs to have the tools to easily manipulate the database structure to add, delete, and modify relations, tables, fields, and properties. All of the data accuracy systems listed above need to be easily manipulated, otherwise an agency risks having to live with outmoded data that cannot be integrated into future analyses. Finally, it is extremely helpful if tools are available to allow restructuring of the data to a new configuration for use in those rare cases when fundamental changes are made.

## 3. DATABASE DESIGN

By setting up a database to meet the needs of data accuracy as described above, many of the problems associated with bad or inefficient design will be eliminated. However, there are a number of other important considerations that still need to be addressed.

### 3.1.1 Meta Data

Meta data are data describing other data. For example, in most databases standard penetration test results are stored. What type of sampler was used? What were the weight and fall of the hammer? Was the hammer driven with a pulley system or hydraulics? Where was the water table relative to the test depth? Each of these pieces of information impacts how much credence given to the N Value results, how N Values from different boreholes and projects can be compared, and how the values can be used in analysis. Such descriptive information is important for good quality data. Therefore, design of the database must go deeper than just what data are needed. It must also specify what data are needed to qualify their values in a meaningful way.

### 3.1.2 Usage Transparency

Another important thing to look for in an efficient database design is whether the fields are "transparent" enough in their labeling so that there is no room for ambiguity. If not, such fields are "opaque" to the user and can invite inaccurate data entry. For example, an empty database might have a field labeled "Water Level." The user might ask, Is this water elevation or water depth? If it is water depth, then is it depth below ground surface or depth below top of casing? This is one example of usage "opaqueness." A transparent database, on the other hand, is unambiguously labeled for both tables and fields.

In addition, commonly there are tables and fields that need more information for clarity than even the best labeling can give. Manuals are necessary, but it is best to strive to make the database self-explanatory. Ideally, either online help or, preferably, on-screen documentation when focus is on those areas would be implemented.

### 3.1.3 Data Granularity

"Granularity" is a term used extensive in the software programming arena, but it also has applicability to the storage of data. The coarser the data structure, the more information is contained in fewer fields. Finer structures break up the information into more fields. "Coarse" granularity gives less information than

a "fine". Coarse data hides its origins. The degree of fineness implemented depends on the needs of each organization.

Coarse-grained database designs are more difficult to check. They also lose information that may be useful in some context in the future or to some other party. Generally coarse-grained database designs originate from designers narrowly focusing on some reporting task, usually a log form, and matching the data structure one-to-one with the final report output.

Following are some examples that illustrate the concept and some of the decisions that need to be made.

### 3.1.3.1 Blow Counts

On many final log forms, the N Value is shown but not the individual blows. A database can be designed with a field just for N Value or with separate fields for each individual 6-inch segment. The former has the advantage of easier input and requires no manipulation for output. The latter reflects the actual collected data and avoids a person performing the N Value calculation. Although that calculation is simple, this is one more opportunity to generate an error, and so it would be safer, from a data accuracy point of view, to input the individual blows and let the reporting tools deal with N Value calculations, including rules on dealing with refusal.

### 3.1.3.2 Component Descriptions

Following are some typical samples of soil descriptions:

Top (m)	Base (m)	Description
0	0.9	Loose dark brown sandy fine to medium SAND with some rootlets (TOPSOIL)
0.9	3.5	Stiff brown with blue veining silty to sandy CLAY with some fine to medium sub-rounded gravel of mixed lithology
3.5	5.5	Stiff brown silty CLAY with some fine to medium sub-rounded gravel and coal fragments, locally a firm, grey silty clay and brown silty fine sand at 4.50-5.00m (GLACIAL DEPOSITS)

Figure 5. Typical Soil Descriptions.

This will be designated as the "blob model" of descriptions, a coarse-grained approach. Following are the same descriptions in a "component model," which comprises a fine-grained approach:

Depth (m)	Base (m)	Strength	Colour	Minor Constituent 1st	Minor Constituents Conjunction	Minor Constituent 2nd	Particle Size 1	Particle Size Conjunction	Particle Size 2	Principle Type	Additional Description	Formation
0	0.9	loose	dark brown	sandy			fine	to	medium	sand	with some rootlets	topsoil
0.9	3.5	stiff	brown with blue veining	silty	to	sandy				clay	with some fine to medium sub-rounded gravel of mixed	
3.5	5.5	stiff	brown	silty						clay	with some fine to medium sub-rounded gravel and coal	glacial deposits

Figure 6. A component model of soil descriptions.

There are many strong advantages to using the component model for data entry:

1. **Consistency** - Consistently structured descriptions are always produced.
2. **Enforced Attribute Selection** - Most of the components would have valid data lists associated with them, from which the user chooses. Such data lists aid in data accuracy and consistency. All of the above components, except for "Colour" (this is a British database example) and "Additional Description," are such lookup lists.

3. **No Formatting** - Formatting is removed from the data. The burden of the formatting is taken from the user and imposed on the software. Further, the formatting can then be altered as needed without changing the data. Note in Figure 6 above that all the components are lower-cased but the final descriptions (Figure 5) are mixed-case. Punctuation is also inserted by user-defined rules. In the above example, all components are separated by spaces (except for the formation which is placed on its own line). However, variable punctuation is possible with commas, periods, colons, etc. placed after any component. Note also the brackets around the formation name. Other component formatting, such as bold, italic, underline, color, etc., are also possible. Finally, the order of the components can be changed at will. User-definable software rules determine the final look of the description.
4. **Selective Reporting** - Not all components may be desirable under some situations. On a log form the full description is usually shown. On a fence diagram showing a number of borehole sticks there is less room, so perhaps only the "Principal Type" and the "Strength" would be shown.
5. **Enhanced Query Capabilities** - Queries can be run on any combination of components. For example, one could ask for N Value results within layers whose "Principal Type" is "sand."
6. **Validation** - Data validation becomes possible. A simple rule could catch "hard sand" or "loose clay."

The level of granularity depends on how the data will be used (what information is important to the final usage of the data) and on how much the organization prizes consistency. Component models can have any number of attributes. They can be as small as three components (principal material, strength, and additional description). The sample above shows only 11 of the 20 actually in the model. Furthermore there is another table for rock descriptions that contains another 12 components.

As an example of how fine this process can go, seven component color models are sometimes used:

Intensity 1	Qualifier 1	Color 1	Color Conjunction	Intensity 2	Qualifier 2	Color 2
light	greenish	red	to	dark	bluish	yellow
medium	pinkish	brown	and	light	grayish	green
very light	reddish	yellow	mottled with	very dark	yellowish	blue

Figure 7. A fine-grained color model.

Each of these components would have a valid value list of allowable choices associated with them.

Some components can be "overloaded." For example, the "Strength" component of Figure 6 has both consistency values for fine-grained soils and relative density values for coarse-grained. This makes the rule for checking if a value is appropriate a bit more difficult to write but, having the rule written certainly makes for easier data entry. Alternatively, one could have two fields: "Consistency" and "Relative Density." This makes the rule writing easier since values cannot exist in both and consistency only applies to silts and clays while relative density only applies to sands and gravels. However, it does make data entry a bit inconvenient in that the user must select the appropriate field for each record.

### 3.1.3.3 Test Result Summaries

In the log report segment in Figure 8, various test results are output for samples in the TESTS AND REMARKS column.

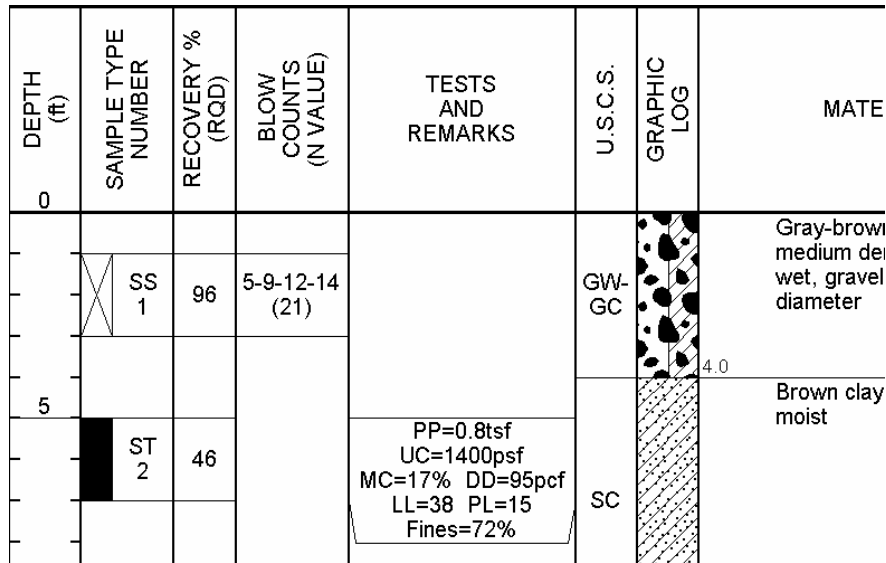


Figure 8. Tests results on a log.

It is common practice to have a field in the database where all of the above results are input, even though the results exist in separate fields elsewhere. This generates several problems including additional input time, more opportunities for mistakes, and a maintenance headache to remember to change the results summary field if one of the component value changes. Results such as these should be taken from the source fields by the report and formatted. Results summary fields should not exist in a database. Like the component model, the burden of the formatting needs to reside on the reporting tool, not the data entry person.

#### 4. DATA COLLECTION

Automatic, electronic recording of laboratory data has been available for many years. In the last 10 years, systems for electronic field logging of borehole logs have become available. Both classes of tools are grossly under-utilized in the profession. This is an area where large efficiencies can be realized with moderate expenditures of resources. Setting up import facilities for reading such data into the database is relatively simple. Electronic data logging eliminates the interface of paper to database transcription and, if appropriately designed, enforces consistency. Proper use of these tools leads to a reduction of errors, less time spent on the collection and data entry phases, and improved data quality.

In the United States it has been a common trend for state transportation authorities to shift more of the data collection tasks onto private consultants. In some cases nearly all data is collected in this way. Where data is obtained externally, electronic submission is essential. Yet it is common for agencies to just require paper copy of data or data in the form of drawings in a CADD format, even though in many cases the consultants have the data in electronic format and the agency also has electronic databases. Inputting the data from paper copy into the Agency's database creates yet another opportunity for error to creep into the records, besides the additional effort that would be avoided with electronic data submissions.

Specifications for electronic data deliverables (EDDs) do not need to specify proprietary formats that require a specific piece of commercial software. They can require something as generic as Excel<sup>®</sup> or ACCESS<sup>®</sup> files that can be generated by readily available tools. In the case of the United Kingdom, most government agencies working with geotechnical data require the submission of data in the AGS data interchange format (AGS Web Site). This is a well supported ASCII file format that can be generated and read by many software packages.

## **5. DATA QUERYING AND REPORTING**

Most organizations, whether public or private, do not take full advantage of their databases. If appropriately designed as discussed above, a database can be accessed to help in the analysis and design of a project. Too often an organization has fixed lists of reports that need to be generated - and the data mining stops there.

There is now a wide range of tools to help query, report, and visualize data. Once they are properly set up, many queries and reports can be run quickly and easily that help engineers understand their sites. Some of the infinite possibilities:

1. Statistics of soil and rock properties.
2. Contouring of ground water and layer boundaries across the site.
3. Plotting of moisture content vs. depth alongside borehole sticks on a fence diagram.
4. Tables of properties filtered on geologic units with mean and standard deviations reported for each property.
5. Correlations of CPT results to strength and consolidation properties.
6. Three-dimensional modeling of various properties.

Many times, what stops organizations from performing these types of analyses is the perceived difficulty and the belief that such analyses require too many resources. However, with proper database design and the proper tools, such analyses can be performed with little effort.

## **6. INTERACTION WITH OTHER SYSTEMS**

One system will never meet all the needs of an organization. Therefore, the database system must have the ability to communicate with other programs and the ability to read and write different file types (Cole 2004).

With any import facility, it is imperative that the data be checked for accuracy first. Ideally the database front end has the ability to implement the various data accuracy checks described earlier and that these same checks are automatically applied to any imported data. If not, an external program needs to be written to check the data prior to import.

Depending on the robustness of a database system, programs can either be written within the system or external to it to perform interactive tasks. Many programs today have object models that can be manipulated by other programs. Therefore, custom applications can be written that access the database to generate spreadsheets, contours, graphs, fences, etc. by handing the appropriate data to applications that expose their functionality programmatically.

In the opposite direction, there are applications that can be pointed to a database to directly generate reports and visualize the data in different ways.

## **7. DATA DISSEMINATION**

An agency may need or want to share its data with other entities. There may also be laws in place that require that its data be made available to the public. Data can be shared in a number of ways:

1. A simple approach is to generate fixed reports to electronic format (usually to PDF).
2. A Web-based GIS application can expose the data visually (VDOT GDBMS).
3. The database can be copied to a standardized format and publicly exposed on the Internet (COSMOS - PEER, Web Site).

Other methods are also possible. The important consideration is that the design of the database system includes consideration of dissemination needs.

## 8. SUMMARY

This paper has reviewed many of the central issues concerning databases in use by transportation authorities. This is by no means an exhaustive survey, nor can it be said that the issues that were addressed were reviewed in depth. The goal, on the contrary, has been to point out potential pitfalls and to expose the industry to useful ideas and different ways of looking at these issues through effective database management.

Appropriate database design has the important benefit of generating usable and maintainable data that can be reported, queried, and shared much more effectively.

No matter how exhaustive and in depth this topic is covered, in the end each agency must apply the available tools to their suit very special needs.

## 9. REFERENCES

AGS: Association of Geotechnical and Geoenvironmental Specialists (United Kingdom). [www.ags.org.uk](http://www.ags.org.uk).

Cole, M. March 2004. Softly does it. *New Civil Engineer*, 32.

COSMOS - PEER: [http://peer.berkeley.edu/year6/yr6\\_projects/ta4/lifelines\\_2L02.html](http://peer.berkeley.edu/year6/yr6_projects/ta4/lifelines_2L02.html).

Hoppe, E. 2003. The use of gINT Software at VDOT. Case study located at [www.gintsoftware.com/case\\_vdot.html](http://www.gintsoftware.com/case_vdot.html).

Plant, G.W. Covil, C.S. and Hughes, R.A. 1998. *Site preparation for the new Hong Kong International Airport*. Thomas Telford Publishing Ltd, London, United Kingdom.

VDOT GDBMS: Statewide VDOT Geotechnical Database Management System (GDBMS) Framework. [http://gis.virginiadot.org/GDBMS\\_menu.htm](http://gis.virginiadot.org/GDBMS_menu.htm).