Developer Guide

SelCtl Land Trust Database Development Project – Client Application

Larry Bednar

June 14, 2004

Purpose

This document describes application modularization used in this project.

Copyright/License

Overview

The application will contain two major components: a “server application” where data resides on a permanent basis, and a “client application” which is installed on computers used by database workers to access and manipulate data held in the “server application”.

The server application will be constructed in MS-Access, MS SQL Server, or in Postgres. The extensive construction of subqueries by SelCtl class modules precludes the use of currently available versions of MySQL, which do not support subqueries.

The client application will be constructed using Microsoft Access 2000 or later using the ActiveX Data Objects (ADO) object model.

Naming Conventions

Naming conventions for data objects (tables, queries, views, columns, etc.) differ from those applied to MS-Access form controls, VisualBasic procedures, functions, classes, properties, etc.

Database Objects

Tables, queries, reports, and forms are named according to these conventions:

1. Individual words or phrases are abbreviated/spelled in the same fashion for every occurrence.
2. Individual words or abbreviations within a name are delimited by underscores

3. Names employ only alphanumeric characters and underscores. No spaces, hyphens, hashes, dollar signs or other special characters are employed. This is done to ensure that the "data server" function can be flexibly moved to other DBMSs at a later time if this should prove desirable.
4. Names must begin with a text character.
5. Names employ only lower case letters.
6. Standard abbreviations are listed in the following table:

Word/phrase	Abbreviation
Definition	defn
Project	proj
Member	mbr
Publication	pub
Columnar	col
Publications	pubs
Report	rpt
Abbreviation	abbr
Identifier	id
Remark	rmk
Amount	amt
Property	prop
Number	nbr
Extension	ext
Yes/no	yn

VisualBasic Objects

Because of the amount of VisualBasic code to be generated, shorter names were found advantageous in code use. As a result, a somewhat different set of naming conventions was applied to objects that were to be referenced primarily in VisualBasic code. In general, VisualBasic names will use shorter word abbreviations, will delimit abbreviations/words within names by capitalizing the first letter (rather than using underscores delimiters), and will use the Reddick naming conventions as outlined in Appendix A of the Access 2000 Developer's Handbook, Volume 1: Desktop Edition.

Form controls and code modules are treated primarily as VisualBasic code objects, with the same basic naming approach as used with VisualBasic variables, etc.

In addition to the Reddick conventions (which cover primarily variable "type" abbreviations used at the beginning of variable names), the following rules are employed:

1. Word/phrase abbreviations are generally standardized throughout VisualBasic code. However, there may be *some* variation depending on the needs of individual functions or procedures. In some procedures, brevity will provide bigger advantages, in other cases, increases specificity may provide bigger advantages. In general, an effort has been made to always use completely consistent naming approaches within individual functions/procedures. There may be slight variations from one code unit to another, however. In general, a word is similarly abbreviated in all places it occurs within database object names.
2. Individual words within a name are delineated by capitalization of the first letter. All other letters in the word use lowercase.
3. The word/phrase abbreviations that are typically used are listed in the table below:

Word/phrase	**Abbreviation**
Selection Control	SelCtl
Selection	sel
Control	ctl
Subform	sfrm
String	str
Combo box	cbo
List box	lst
Text box	txt
Label	lbl
Recordset	rst
Form	frm
Object	obj
Long ("long" variable type)	lng
Number	nbr
Column	col
Command button	cmd
Record	rec
Report	rpt
Application	app
Duplicate	dup
Value	val
Boolean	b

Word/phrase	Abbreviation
Integer	int
Specification	spec
Subquery	sqry

Module Descriptions

This section contains descriptions of code modules planned for use in the client application. Modules are described in terminology appropriate to MS-Access.

Code Modules

This section contains descriptions of all public code modules to be provided in the new system. These modules will provide functionality used by multiple forms/code modules in the final application.

Common SelCtl Class Module Characteristics

All SelCtl class modules contain a number of methods and properties of similar function. While internal implementations differ according to the specific needs of each different class, the overall function of these methods and properties are accessed in similar fashions, and the products of these actions are used similarly:

CtlAssoc – When passed a object variable for a control, returns a Boolean indicating whether the control is associated with the SelCtl object.

Clear – Sets stored and current selection expressions and descriptions to blank, and resets the displayed value of associated form controls to "blank".

RevertCtlValue – Sets associated control to a value previously stored by the SelCtl object

RevertSelValue – Sets the SelClause value associated with the SelCtl object to a value stored earlier.

SelClause (Property Get) – Provides the current value of the SelClause associated with the SelCtl object. This value contains an SQL expression that may be inserted into an SQL SELECT statement to select matching rows in the return set of the SELECT statement.

SelDesc (Property Get) – Provides the current value of the SelDesc associated with the SelCtl Object. This value contains an English-like description of the conditions imposed by the control associated with the SelCtl object.

SetProperties – Guides the user in specification of all properties required of the SelCtl object.

SetSelClause – Constructs an SQL expression from the control associated with the SelCtl object.

SetSelDesc – Constructs an English expression from the control associated with the SelCtl object.

SetStoredCtlValue – Stores the current control value in case the user decides to later revert a new set of criteria.

SetStoredSelValue – Stores the current SelClause value in case the user decides to later revert a new set of criteria.

TargetCol – Name of column targeted by the conditions specified by the user using the control associated with the SelCtl object.

TargetTableAlias – Alias to be used for the table containing the target column.

Code Module - basSelCtlShared

This module contains functions, procedures and variables/properties that are used by multiple "record selection" forms. This modularization approach was adopted in an effort to reduce overall code maintenance effort, since maintenance of a single shared copy of code should be easier than synchronization of multiple copies of similar code. The more complex procedures needed for operation of "record selection" – originally located within each of the form modules themselves, are now located in this code module.

Code Module – basShared

This module contains functions, procedures and variables/properties that are used by multiple "record selection" forms *as well as* by SelCtl class modules, etc. This modularization approach was adopted in an effort to reduce the amount of code maintenance effort, since maintenance and synchronization of multiple copies of similar code should be expected to increase maintenance effort.

Code Module - InitApp

This module downloads lists of all static data used to populate combo and list box controls, etc. This is done to prevent frequent queries from client to data server during form use, etc. for purposes of enhancing client/server performance.

Procedure "DownloadServerData" – This procedure downloads data used repeatedly in lookups, control row sources, etc. from server to the client application at application startup. The purpose is to improve performance by downloading static, unchanging data that is used repeatedly by the client application a single time, rather than passing data from server to client each time this data is requested. Data targeted for this type of transfer will typically be code definitions.

Module – "SelCtlCbo" Class Definition

This module defines methods and properties associated with combo box controls used on "record selection" forms. The class is constructed to allow use with number, date, or text values according to user specification of object properties.

Code Module – "SelCtlGrp" Class Definition

This module defines methods and properties associated with controls used on "record selection" forms to select records related to "group membership". Methods required for this purpose differ somewhat from those used with other "SelCtl" objects because of the simultaneous use of "start date", "end date" and "effective date" parameters to determine group membership. This contrasts strongly with all other SelCtl objects (other than SelCtlObjList objects), which all are associated with only a single form control.

With regard to the structure of the multi-select list box provided to allow the user to select groups of interest, this class assumes that column 0 of list box contains numeric id value(s) to be used in filter expression, and that column 1 of list box contains descriptive phrases more readily understandable by application users.

Code Module – "SelCtlList" Class Definition

This module defines methods and properties associated with multi-select and single-select list box controls on "record selection" forms.

Assumes that column 0 is contains the values to be used in filter statements.

Code Module – "SelCtlObjList" Class Definition

This module defines methods and properties for working with a complete set of SelCtl objects defined for use with a "record selection" form. In general, the "record selection" form passes generic commands to an object in this class. The SelCtlObjList then determines how to access properties and methods of appropriate individual SelCtl objects and make use of it's own methods and properties to construct the information required by the "record selection" form.

ActiveSelCtlObj – Returns an object reference to the SelCtl object associated with the provided control reference.

AddRelatedTable – Guides developer in addition of an array element to an internal array used to manage use of information about tables involved in selection, but not a part of the base table list for the SQL SELECT statement being used by the associated "record selection" form.

AddSelCtl – Guides developer in addition of an array element to the internal array used to manage use of information about SelCtl objects defined as a part of the SelCtlObjList object.

AddToRelatedTableSqry – This procedure builds a complete SQL SELECT subquery when provided the alias of a table represented in the mart array and an SQL expression to add to the subquery associated with that table alias.

AddToRelatedTableSqryDesc – This procedure builds a human-language description of a subquery provided the alias of a table represented in the mart when array and an expression to add to the subquery associated with that table alias.

BaseJoinDesc (Property Get) – Retrieves value of SelCtlObjList object property holding join description portion of SQL SELECT statement used by "record selection" form

BaseRowDesc (Property Get) – Retrieves value of SelCtlObjList object property holding row description portion of SQL SELECT statement used by "record selection" form

BaseTableList (Property Get) – Retrieves value of SelCtlObjList object property holding table list portion of SQL SELECT statement used by "record selection" form

CheckSelCtlChange – Checks count of rows selected after a control change, reverts control value if count is zero, sets row count displayed on form appropriately.

Clear – Runs the "clear" method for each SelCtl object associated with the SelCtlObjList object.

RelatedTableIndex – Returns the array reference related to the table alias passed to the method.

RevertObjVal – Determines which SelCtl object is associated with the specified control and runs the RevertCtlVal method for that object

SelDesc (Property Get) – Retrieves value of property from SelCtlObjList object –based on combination of results from *all* SelCtl objects defined in the SelCtlObjList object.

SelFilter (Property Get) – Retrieves value of property from SelCtlObjList object –based on combination of results from *all* SelCtl objects defined in the SelCtlObjList object.

SetObjSelClause – Determines which SelCtl object is associated with the specified control and runs the SetSelClause method for that object

SetProperties – This method is designed to faciliate entry of properties required during use of the class, and to ensure that the developer is correctly prompted for all properties required

SetSelDesc – Set human-language description of overall filter expression

SetSelFilter – Constructs filter expression corresponding to complete list of SelCtl objects. It is expected that this procedure is run in response to a before update or after update event on a form.

SetStoredObjVal - Determines which SelCtl object is associated with the specified control and runs the SetStoredCtlVal method for that object.

Code Module - SqryMList

This module defines methods and properties associated with multi-select list box controls on record selection forms in cases where they are used to select records based on existence or non-existence of records in related tables tested using SQL subqueries.

Code Module – "SelCtlTxt" Class Definition

This module defines methods and properties associated with use of text box controls on record selection forms.

Code Module – "SelCtlTxtPattern" Class Definition

This module defines methods and properties associated with text box controls used on record selection forms to specify text-matching patterns used to select records.

This module includes features to appropriately deal with "special characters" that might be part of a text string (apostrophe, quote, etc.).

Form Modules

This section contains descriptions of all forms to be provided in the new system.

In general, all edit forms provided as part of the application will employ SQL statements assigned to the "data source" property. This approach will ensure that no queries, etc. are exposed to the user in a way that might allow alteration, etc. The user will therefore have the "query" area of the client application completely available for their own use in defining/storing queries.

All forms will reside in the "client" portion of the application, leaving choices of DBMS, etc. for the "data server" portion of the application the maximum possible flexibility.

Forms will derive all possible data used for populating combo boxes and list boxes from local tables populated by download from data server at start-up of client side application.

Form – Common "Record selection form" characteristics

Most supported tasks in the new system will use a "record selection form". Many of the characteristics of these forms are common. Those characteristics held in common are described in this section.

In "data editing" tasks this tool will perform the following functions:

1. Allows the user to efficiently search for a record to be edited
2. Facilitates users' systematic checking for pre-existing records to prevent entry of a possible duplicate record.
3. Reduces the amount of data passed between data server and client interface application – facilitating high performance of the system by avoiding unnecessary data transfers across network.

In "reporting" tasks this tool will perform the following functions:

1. Allows the user to efficiently select the set of records to be represented in a report, and exported data file, etc.

Forms of this type will provide a variety of unbound controls (no bound controls will be used) facilitating the user's specification of the most commonly used selection criteria.

All controls for these selection criteria are declared to be members of one of the "selection control" classes defined in the application. Select control objects include defined properties and methods that facilitate construction of SQL expressions used as MS-Access filter clauses to implement the selection criteria indicated by the user.

After the user updates each control, the form constructs a query to the server database that returns the count of records matching the revised selection criteria. If the count is zero, the user is offered an opportunity to revert the control to the values in effect previously. The count is displayed in a read-only control on the form for the user's benefit. This information should be of assistance to a user who wants to know whether to add additional selection criteria to pare down their selection list further.

The form also provides the following command buttons:

1. "Display matching records" – Typically, this moves the user to a related "selected records display form" where a concise display of information from records matching the user-specified selection criteria is displayed.
2. "Exit" – Sends the user back to the main switchboard. (Executes "ExitRecSelectionForm" procedure.)
3. "New selection" – Deletes all selection criteria set by user and places them in a suitable position to start entering a new set of selection criteria.

Form – Common "Selected records display form" characteristics

Most supported tasks in the new system will employ a "selected records display" form. Many of the characteristics of these forms are common. Those characteristics held in common are described in this section.

All controls provided on the form are unbound.

This form provides the user a compact display of the records selected by their entries on the associated "record selection form". The display is provided in a list box control (**lstRecsDisplay**) using a data source property into which the filter clause constructed by the associated "record selection form" has been inserted.

In data editing tasks, this form provides the following functions:

1. Displays an abbreviated list of records matching characteristics entered by the user.
    a. This facilitates the user's selection of a single record for editing.
    b. This allows the user to quickly check whether a record representing the information of concern is pre-existing in the system. This should help reduce duplicate entries of information.
2. The **lstRecDisplay** control is set up to allow only a single record to be selected.

In reporting tasks, this form provides the following functions:

1. Allows the user to review the set of selected records to verify that they are the correct set to include in a report.

2. The user is provided the capacity to indicates the type of output to be generated, and the system produces the desired report.

The form will also provide a mechanism for the user to select a "report/output" method to be used in summarizing the selected records. The options provided will depend on the specific task being supported by the form. To preserver screen space and clarity, a combo box may be the best choice for displaying possible user selections. It may be desirable to alter the display of "selected records" according to the "report" option selected by the user.

The user will be allowed to select the following actions:

1. "Exit" – sends the user back to the main switch board
2. "Revise selection" – sends the user back to the associated "record selection form" to revise the user-specified selection criteria. (Executes "OpenRecSelectionForm" procedure.)
3. "Output format" – displays a list of the available output formats in a combo box, enabled only if the user has indicated "report" functions should be performed.
4. "Report" – performs the reporting action specified by the user (enabled only if the user has indicated "report" functions should be performed)
5. "Edit selected record" – Opens the record selected by the user for editing (enabled only when the user has indicated "edit" functions should be performed)
6. "New record" – Creates a new record and prompts the user for values (enabled only when the user has indicated "edit" functions should be performed)

Form – Common "record edit form" characteristics

Main data subjects in the system are accessed for editing using these forms.

These forms generally display attributes of a single record in a primary subject area. They typically use subforms to display data from tables related to the main table for the user's edit/review.

Controls for all attributes allowing user-specification are provided. Controls that allow user entries are generally "sunken" and use a white-colored background. Fields that require user entries are additionally labeled with bold text. Controls that do not allow user entry are generally "flat" and use a gray-colored background.

The user will be allowed to select the following actions:

1. "Exit" – sends the user back to the main switch board
2. "Delete" deletes the selected record.
3. "*Subject* display" – Returns the user to the associated "record display" form.

Form – Code Definition Edit Contact Method
Datasheet style form for editing records, for use only by administrators.

Form – Code Definition Edit Contact Subject
Datasheet style form for editing records, for use only by administrators.

Form – Code Definition Edit County
Datasheet style form for editing records, for use only by administrators.

Form – Code Definition Edit Donation Type
Datasheet style form for editing records, for use only by administrators.

Form – Code Definition Edit Fund
Datasheet style form for editing records, for use only by administrators.

Form – Code Definition Edit Group Type
Datasheet style form for editing records, for use only by administrators.

Form – Code Definition Edit Location Type
Datasheet style form for editing records, for use only by administrators.

Form – Code Definition Edit Party Category
Datasheet style form for editing records, for use only by administrators.

Form – Code Definition Edit Phone Type
Datasheet style form for editing records, for use only by administrators.

Form – Code Definition Edit Region
Datasheet style form for editing records, for use only by administrators.

Form – Code Definition Edit Related Person Role
Datasheet style form for editing records, for use only by administrators.

Form – Main Switchboard

Provides user the option of switching to any of the "record selection" forms in the system, or to the "code definition" forms.

Report Modules

This section contains descriptions of all reports designed for use in the new system. Many of these will be used by several other forms/code modules in the system.

All reports will be defined as part of the "client" portion of the application.

All "data source" properties will be set using SQL statements assigned to that property. Stored queries that will be visible to the user within the MS-Access "query" collection will be avoided.

*Report – Contacts

This report displays a list of contact records.

Applies a filter clause supplied by the calling module.

Information displayed includes:

- Donor relationship name
- Staff member name
- Contact subject(s)
- Notes from the contact
- Contact date

Report – Easement Project Landowner

Displays information about parties involved in easement projects in the role of "landowner".

Displayed information includes:

- Easement project
- Property name

Easement project manager

Name of primary staff contact

Party name

Name of primary contact for the party

Name of spouse for the primary contact

For all locations associated with the "party":

Location type

Street address

City

State/Province

Country

Postal Code

Phone numbers associated with the location

Phone numbers associated directly to the "party" rather than to specific locations

Name of primary staff contact

All "party category" codes assigned to the "party"

All "groups" to which the "party" belongs at the time the report is generated. [LB – Since we expect groups to replace many of the current "status codes", it seemed like these should be included.]

Property manager/caretaker [LB – I'm not sure we've explicitly included a spot for this information in the current data model. I suppose the caretaker is a "party" – someone who's address, email, etc you'll want to track. And that a "easement_proj_party" record should be entered linking that "party" to the easement project and indicating the role of "caretaker" or "property manager".]

The displayed information is arranged by easement project

Report – Donation/Budget Comparison

*Report – Group Members

This report displays a list of parties belonging to a specified group as of a user-provided "effective date". I proposed this mostly an on-screen group membership review tool for database users.

Information displayed includes:

Relationship ID

Relationship name

For primary relationship contact:

First name

Last name

Membership start date?

Membership end date?

*Report – Land Steward Easements

Displays a list of all easements assigned to each land steward..

Information displayed includes:

Easement project ID

Land steward name

Name of primary staff contact for the landowner involved in the easement [LB – Have I got this right? Or is it the "easement project manager" that you'd be more likely to want?]

Locations

*Report - Locations

Displays a list of all locations.

Information displayed includes:

For each location:

Location ID

Attributes of location (address, city, etc.)

The name of the primay contact for each “party” associated with the location

Report – Mail Merge File

DOS file containing information required to perform a basic letter merge for a select group of individuals. Set up specifically to feed information to MS-Word letter merge operations. The information displayed characterizes a group of relationship/persons matching a set of selection criteria specified by the user:

Information used will include:

Relationship name

For primary relationship contact:

First Name or Nickname?

Last Name

Name prefix

For primary location:

Street Address

City

State/Province

Postal Code

*Report – Mailing Label

Generates mailing labels for party records.

Uses Avery 5162 address labels for printing.

Applies record filter constructed by calling application. Filter should always include provisions to use only the primary contact location for the party.

Includes the following attributes:

Party name

Street address

City

State/Province

Postal code

Country

Desirable to automatically insert bar code representing postal code on labels.

[LB – Is there a specific ordering of selected records that is desirable?]

*Report – Location File

This will be a DOS file that contains the information needed for a mailing service to perform a mailing assembly operation. The file will be a DOS text file with tab-separated fields.

Information used will include:

Relationship ID

Relationship name

Street Address (for primary contact location)

City (for primary contact location)

State/Province (for primary contact location)

Postal Code (for primary contact location)

Country (for primary contact location)

*Report – Address File

This report provides information that might be exported from the database for a staff member's inclusion in an address book or similar.

Information used will include:

Party name

Name of primary contact person for the party, including:

First Name or Nickname of primary contact person for the party

Last Name of primary

Name prefix

Street Address

City

State/Province

Postal Code

Primary email address

Primary phone number

*Report – Donation Detail

This report lists information about each donation. The donations are ordered by party and then by donation date.

Information displayed includes:

Donor party name

For each donation matching the selection criteria:

Donation date

Amount

Pledge (yes/no)

Pledge amount

Proxy donation (yes/no)

Proxy donation original donor relationship name

Fund contributed to

Sum of “amount” values

Sum of “pledge amount” values

*Report – Party Donation Summary

(Very similar to the Donation Detail report, but with concise summaries displayed for each individual rather than detailed displays of each donation.)

This report lists summary information for all donations made by parties. The displayed information is ordered by party name.

Information displayed includes:

Donor relationship name

For each fund contributed to:

Fund(s) to which the individual/relationship donated during the time period

Total amount contributed to the fund

Improvement Ideas

The following improvement ideas are being considered for implementation during further development:

1. Enhance current error checking for multiple "primary" entries for party location, phone, email to automate a change of a previously entered "yes" value to a "no" value after prompting the user for confirmation. (Current version only tells user than another related record already has a "yes" value for the current party, then cancels the addition of a second "yes" to the set of records.)
2. Add more detailed editing capacities for "donations" tab of "party_edit" form. Ability to switch from "list" to "columnar" forms might provide the ability to edit/complete donations directly from "party_edit" form.
3. To enhance performance of report generation, create and populate local tables corresponding to data used in reports. According to MS-Access documentation, this might result in enhanced report generation performance since Access report functions are repeatedly access data during report generation. Complete transfer of data involved in reports in a single operation sometimes provides superior performance to repeated data withdrawals from the data server.
4. Change yes/no combo box controls from current single-column, yes/no format to two column format with explicit value list (0;No;-1;Yes). This will prevent display of actual underlying 0/-1 values when user clicks into field (current behavior).
5. Provide capacity to switch to different summary displays of attributes in list box of "record selection display" forms. In some cases, alternate groups of attributes may be useful to the user.
6. Add ability for user to rerun downloading of code definitions, etc. which may on some occasions be changed during the duration of a user's work session with the "client" application.

7. Create a tab on the main switchboard for individual reports. Buttons on this tab should correspond to individual reports. When clicked, each of these buttons should open the appropriate "record selection" form for the report, while pre-selecting the report, disabling "edit" and "new" buttons, etc.

References

Getz, Ken, Paul Litwin, & Mike Gilbert. Access 2000 Developer's Handbook Volume 1: Desktop Edition. 1999. Sybex, San Francisco. 1613 pp.

Litwin, Paul, Ken Getz, & Mike Gilbert. Access 2000 Developer's Handbook Volume 2: Enterprise Edition. 2000. Sybex, San Franscisco. 1071 pp.

Novalis, Susann. Access 2000 VBA Handbook. 1999. Sybex, San Francisco. 845 pp.

Roman, Steven. Access Database Design & Programming. 2d Edition. 1999. O'Reilly, Cambridge. 409 pp.