

Using BLOBs to Track Edit History of Water Right Features

Stephen N. Hayes, Liz Ayarbe, and Christina Noftsker

New Mexico Office of the State Engineer (NMOSE)

June 18, 2004

ABSTRACT:

The NMOSE has developed a technique using BLOBs (Binary Large Objects) to maintain and perpetuate documentation relating to the delineation and editing of spatial water right features. Feature-level metadata is necessary due to complex requirements involved in the creation and maintenance of water right geospatial data. Notes and comments pertaining to the edit history of each individual spatial feature are stored in a BLOB field within the feature class attribute table in an ArcGIS Geodatabase. Hydrographic Survey Specialists working with water right feature layers use a customized utility, the Water Right Edit Journal (WREJ), to view feature documentation and make additional entries while editing features. Direct association of feature metadata with individual features has simplified maintenance of documentation as well as promoted the tendency to make comment entries, resulting in over-all better feature documentation. In addition, inheritance of feature documentation during edit procedures such as feature splitting is automatic and provides an integrated method for tracking feature lineage. WREJ requires a minimal amount of VBA scripting and can easily be adapted to other BLOB applications.

INTRODUCTION AND BACKGROUND:

The New Mexico Office of the State Engineer (NMOSE) is responsible for the administration and adjudication of the state's surface and ground water resources, involving 266,000 water right files within thirty-three administrative basins. Water right data, like cadastral data, are complex and dynamic. As with management of any complex data, metadata is required for feature classes (layers) and individual features. One of the roles of the NMOSE is to track and maintain changes such as water right transfers and changes in water use for administrative and adjudication purposes. The ability to track feature lineage becomes increasingly important as a spatial dataset grows and matures, especially when frequent edits are necessary. The legal determination of water rights, known as an adjudication, can last anywhere from 2 to 10 years. While an adjudication is in process, the supporting geospatial data, known as a hydrographic survey, has to be updated and maintained so that the most current information regarding ownership and water right status is constantly and continuously reflected by the geospatial data.

With recent advances in GIS technology, the most significant being the advent of the ESRI ArcGIS Desktop environment, the ability to edit geospatial data and maintain currency has increased dramatically, resulting in new demands being placed on Hydrographic Survey Specialists charged with maintaining this data. Previous attempts to annotate edit comments have amounted to an array of partially populated and excessively long text fields in feature class attribute tables. Hydrographic Survey Bureau (HSB) members have noted that standard text field width (250 characters) is too small for the volume of text information required to make sufficient edit comments, resulting in incomplete and often unintelligible feature documentation. Such deficiencies have led to

problems in bringing new HSB staff up to speed on the history of a given hydrographic survey. In light of this problem, one recent attempt to fully document water right features utilized a Microsoft Access database to store edit comments in a Memo field. However, the user interface for this external database wasn't completely integrated into the ArcMap environment, resulting in a lack of acceptance, and consequent failure of utilization among HSB staff. Again, the search for a sound method to successfully document water right features was renewed, resulting in the discovery of the ability to store edit comments, along with important ancillary information such as editor's name and time/date stamp for each entry, in a fully-integrated manner using a Geodatabase BLOB, or Binary Large Object, field.

Within the ESRI Geodatabase data model, the BLOB data type is utilized in a variety of ways, ranging from the storage of geographic coordinate data as Smart BLOBs (points, lines, polygons, etc.) to the storage of feature class and dataset metadata files in XML format. Even specialized annotation layers are reduced to BLOBs for storage in BLOB fields. The fact that the BLOB data type has become so prevalent within the Geodatabase model prompted the initial investigation into using this highly adaptable data type to store edit comments. Comments pertinent to transfers of ownership, along with field notes, historical notes, and comments regarding legal status are entered into a text document and stored in a BLOB field attached directly to the feature class attribute table with a customized utility, the Water Right Edit Journal (WREJ). The following discussion focuses on the maintenance of the feature metadata BLOB field in the feature class attribute table instead of focusing on the WREJ application itself. Examples and illustrations are drawn from the WREJ interface and its VBA source code. This approach will provide guidance for developers wishing to utilize the BLOB data type for feature metadata tracking as well as other purposes.

BLOB BASICS:

A BLOB is any data in a digital binary form. One definition of "binary" is that for a given condition, there are two and only two states. This concept can be likened to a light switch that has two positions, "on" and "off". In the context of computers, the binary form is represented by the bit, which is a single location or piece of information having a value of either 1 ("on") or 0 ("off"). The bit is the simplest unit for storage and communications in the world of computers. In this light, anything that exists in the digital realm, regardless of current format (.doc, .xls, .ppt, .jpg, etc.) and location (in physical memory or written on disk) is in binary form. There is no difference technically between a digital entity (document, image, etc.) being in binary form or existing as a Binary Large Object. The key distinction is BLOBs exist solely in databases.

The concept of the BLOB is a product of the rise of Relational Database Management Systems (RDBMS). Databases are designed to manage, maintain, and store data through the use of pre-defined data types. Data types familiar to the GIS user are integer, floating point (decimal), character (text), etc. While the BLOB data type is relatively new to GIS, it has existed within the realm of RDBMS for more than 20 years. Only recently has the BLOB data type made its way into GIS through the Geodatabase data model, which employs database management systems such as Oracle, DB2, SQL Server, and Access to maintain spatial data. A database BLOB field can be thought of as a blank piece of memory onto which any amount of data, in binary form, can be written.

The only catch is that once a piece of data has been stored as a BLOB in a database BLOB field, it becomes basically inaccessible from outside the RDBMS environment.

One of the major deterrents to utilizing the BLOB data type within databases in the past was that customized applications were required to read, or “load”, data stored in a BLOB field for access by the user and then write, or “store”, the data back into its binary resting spot. Indeed, the database user cannot determine by looking at a database table whether or not a BLOB field has anything in it. When a feature class attribute table is opened in ArcGIS via a Table View, all that the GIS user sees in each row of the BLOB field is a single word, “Blob” (Figure 1).

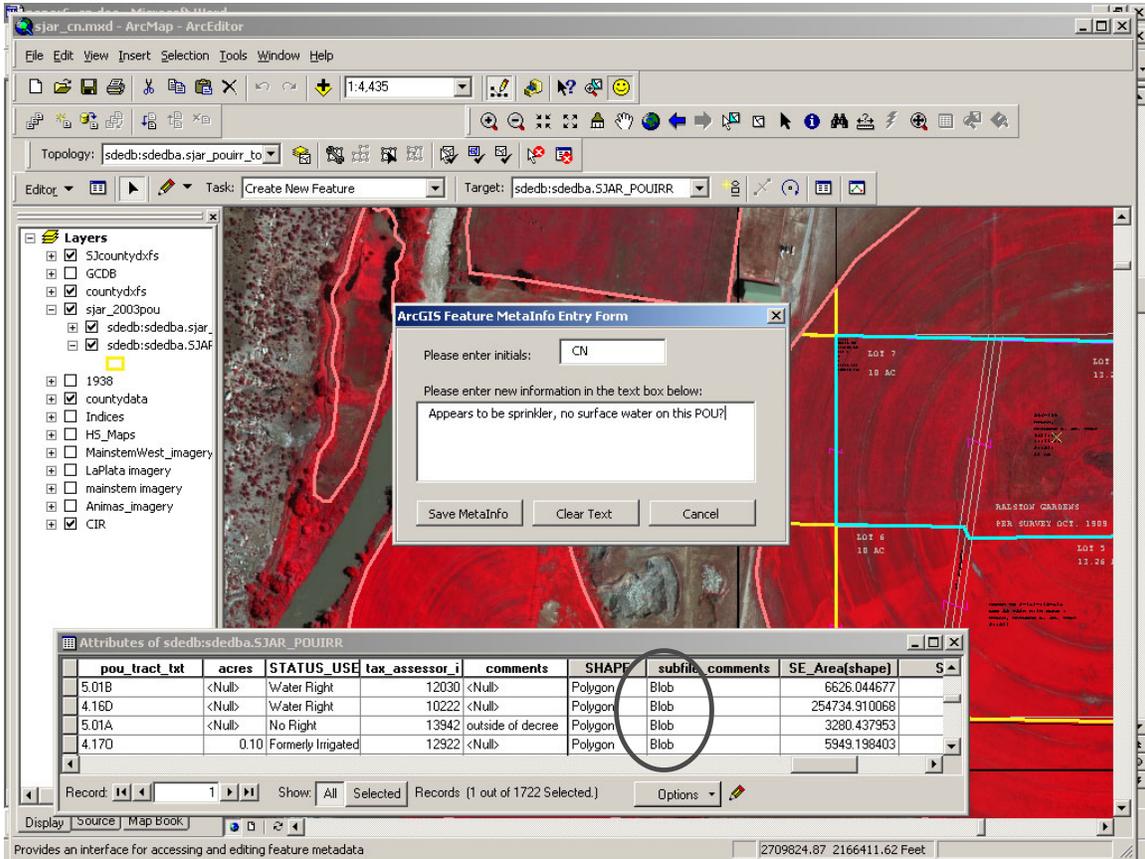


Figure 1: Depiction of the contents of the BLOB field "subfile_comments"

The WREJ entries are not viewable within the attribute table’s BLOB field. This has been the source of much concern over where the data that is supposed to be in the BLOB field actually is. Consider the “Shape” field in any Geodatabase feature class attribute table: knowing that the coordinate data for each individual feature is stored within this field as a BLOB, should there be concern that only the words “Polygon”, “Line” or “Point” are shown in this field when the contents of the table are viewed? With regard to the shape field, ArcMap is the customized application that reads and writes the coordinate data from this field and makes that data available to the GIS user. In the same manner, WREJ is the customized application that allows the Hydrographic Survey

Specialist to view and make entries into the Edit Journal for each feature. Written in relatively easy-to-understand VBA (Visual Basic for Applications) script (as opposed to Fortran, 4GL, etc.), WREJ puts the ability to utilize the BLOB data type within the grasp of GIS users.

CREATING AND STORING BLOBS:

Anything in digital form can be stored in a database BLOB field. In the case of WREJ, an ArcMap Text Element (Label) is stored in a BLOB field called “subfile_comments”. The flexibility and adaptability of working with BLOBs in the ArcMap environment is demonstrated by utilizing the Text property of the Text Element object in the present WREJ application. This concept can be adapted to work with any entity included in the ArcObjects object model. The ability to store an object in a BLOB field is achieved through a two-step process of obtaining the digital form of that entity and storing that binary form in a BLOB field. Using VBA, WREJ performs this task using the segment of code given below (Figure 2).

```

If newInfo = "" Then
    MsgBox "No new information entered into the MetaInfo" & vbCrLf & _
        "BLOB for the selected feature", vbInformation, "ENTRY NOT STORED"
    pEditor.AbortOperation
    Exit Sub
ElseIf eFlag = 1 Then 'if BLOB exists, then append; otherwise, create
    pTextElement.Text = pTextElement.Text & vbCrLf & vbCrLf & dateStamp & _
        vbCrLf & newInfo
    Else
        pTextElement.Text = dateStamp & vbCrLf & newInfo
    End If

'Persist the new text label element to memory
Dim pPersistIn As IPersistStream
Set pPersistIn = pTextElement
Dim pMemStreamIn As IMemoryBlobStream
Set pMemStreamIn = New MemoryBlobStream
pPersistIn.Save pMemStreamIn, False

'Save the new MetaInfo BLOB to the BLOB field for the specified feature
pRow.Value(pRow.Fields.FindField("subfile_comments")) = pMemStreamIn
pRow.Store

'Stop Edit operation for BLOB edit

```

Figure 2: Storing an ArcMap Text Element in a BLOB field

In Step 1, the Text Element is persisted to physical memory using the Visual Basic IPersistStream interface. This allows access to the binary form of the Text Element, which can then be converted to a new BLOB. After the BLOB containing the Text Element in binary form has been created, it can then be stored in the BLOB field, as shown in Step 2. Saving the new BLOB into the “subfile_comments” field in this manner over-writes any existing data in the field. Given this logic, the only change required to store something other than a Text Element in the BLOB field would be in Step 1, where the Text Element is persisted to memory. However, other changes to the application would be required to determine how the user interacts with whatever is retrieved from the BLOB field. Considering that WREJ is designed to accept user input in a text form, the majority of the programming code in its VBA script pertains directly to

managing this input. If the object or data stored as a BLOB were to be used in a non-interactive manner, such as for the simple display of an image file, the small amount of scripting would be further reduced.

EDITING AND MAINTAINING AUXILIARY BLOBS:

At the onset of this development effort, it was discovered that the additional BLOB field could not be maintained within a Geodatabase feature class attribute table without first isolating the ArcMap Edit Operation that allows access to WREJ BLOB. Defining an Edit Operation (Figure 3) creates a separation of edits to the WREJ BLOB from edits to the feature coordinate data. When an ArcMap Edit session is initiated and a feature class is opened for editing, the Shape BLOBs for any selected features are automatically opened and read, thereby allowing access to the feature coordinate data. During initial attempts to open and access the WREJ BLOB for a selected feature within an active Edit session, the Shape and WREJ BLOBs could not be opened for access simultaneously within the same Edit Operation. After isolating the WREJ Edit Operation from the rest of the active Edit Session, there were no further conflicts.

```
Normal.mxt - ThisDocument (Code)
UIButtonControl4 Click
If pField.Type <> esriFieldTypeBlob Then
    MsgBox "Field 'subfile_comments' in " & pFeatLayer.Name & _
        vbCrLf & "is not a BLOB field", vbExclamation, _
        "FIELD TYPE ERROR: CANNOT PROCEED"
    Exit Sub
End If

'Make sure that at least one feature is selected
If pSelSet.Count < 1 Or pSelSet.Count > 1 Then
    MsgBox "Please select one and only one feature", vbExclamation, "FEATURE SELECT"
    Exit Sub
End If

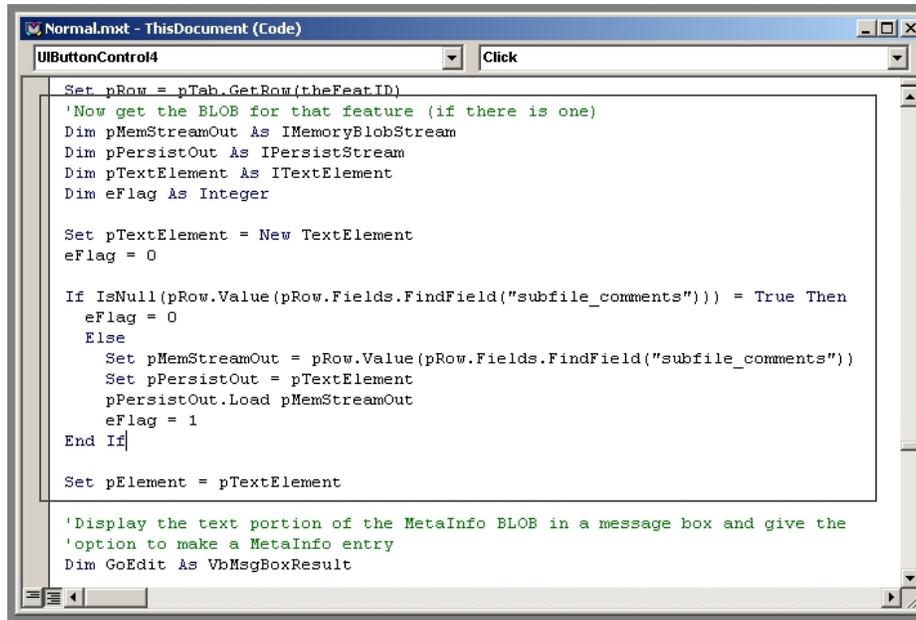
'If in Edit mode, then isolate BLOB access
If pEditor.EditState = esriStateEditing Then
    pEditor.StartOperation
End If

'Grab the FC attribute table row based on selected feature ID
Dim pRow As IRow
Set pRow = pTab.GetRow(theFeatID)
'Now get the BLOB for that feature (if there is one)
Dim pMemStreamOut As IMemoryBlobStream
Dim pPersistOut As IPersistStream
```

Figure 3: Isolating the WREJ BLOB Edit Operation

Once access to the WREJ BLOB for a selected feature has been properly isolated, the BLOB can then be loaded from its BLOB field and used to reconstruct the Text Element that stores the feature edit comments and notes. This is basically a reversal of the process used to store the Text Element into a BLOB field, but has been illustrated below for the sake of completeness (Figure 4, see below). Notice in this code segment that a new Text Element is created to house the Text Element that will be read from the WREJ BLOB. The new Text Element is persisted to memory and then loaded with the BLOB contents of the “subfile_comments” field. Also notice the “If...Then” loop designed to test whether or not the BLOB field for the selected feature already has

existing WREJ entries. This logic is necessary if BLOBs are to be created on-the-fly as opposed to priming the BLOB field for each and every feature with BLOBs containing pre-determined or “root” information.



```
Normal.mxt - ThisDocument (Code)
UIButtonControl4 Click
Set pRow = pTab.GetRow(theFeatID)
'Now get the BLOB for that feature (if there is one)
Dim pMemStreamOut As IMemoryBlobStream
Dim pPersistOut As IPersistStream
Dim pTextElement As ITextElement
Dim eFlag As Integer

Set pTextElement = New TextElement
eFlag = 0

If IsNull(pRow.Value(pRow.Fields.FindField("subfile_comments"))) = True Then
    eFlag = 0
Else
    Set pMemStreamOut = pRow.Value(pRow.Fields.FindField("subfile_comments"))
    Set pPersistOut = pTextElement
    pPersistOut.Load pMemStreamOut
    eFlag = 1
End If

Set pElement = pTextElement

'Display the text portion of the MetaInfo BLOB in a message box and give the
'option to make a MetaInfo entry
Dim GoEdit As VbMsgBoxResult
```

Figure 4: Loading BLOB data into an Object

For the purposes of WREJ, if there are previous WREJ entries for a given feature, the BLOB is loaded into the new Text Element. If the “subfile_comment” field for the selected feature is empty (IsNull), then nothing is done with this field until the new WREJ entries are saved which initiates the process in Figure 2.

CONCLUSIONS AND CAUTIONS:

Using the techniques outlined in the previous discussion, a GIS analyst can create, edit, and maintain auxiliary BLOB data stored in direct association with Geodatabase feature class data. For the OSE, this has provided a means of storing and maintaining feature documentation for very dynamic spatial data. In light of the flexibility of both the BLOB data type and the techniques outlined here for storing and accessing BLOB data, the use of BLOB data within feature class attribute tables is likely to expand into other applications, both more sophisticated and simpler in nature. Plans are being formulated for expanding the WREJ application into a full-featured feature lineage tracking system that will allow those who maintain these data to determine, with simple inspection, the origin and evolution of any given water right feature. This will aid not only in maintaining information about why certain edits were made, but ultimately will provide a readily-accessible means of tracking how water use, as associated with water rights, changes over time. This very aspect is one of the most elusive factors in the administration and determination of water rights, and at the same time is inherent to the very nature of water use.

Indeed, the use of auxiliary BLOBs in feature class attribute tables may provide a way of making spatial features “smarter”. Given that information regarding the actual

spatial feature, or even its attributes, can be stored in an ancillary manner in an auxiliary BLOB field, such “MetaInfo” can be read, interpreted, and updated in an automated way. Thus, existing spatial data could be enhanced for use by someone attempting to understand and make sense of that data. For instance, when a feature class is brought into an ArcMap project, information on how to display and provide accessibility for that feature class will be read from the auxiliary BLOB for each feature. Gone will be the days of trying to interpret attribute codes in order to determine symbology for a desired map. No longer will GIS analysts have to argue over the edit status of features in a given feature class, or spend time trying to determine what the origin of a certain polygon is. All this information will be stored and maintained in a BLOB directly attached to the feature class attribute table as feature MetaInfo.

Caution must be exercised in developing plans for using auxiliary BLOB data. In the effort under way at the OSE, development has proceeded very cautiously considering that this is relatively uncharted territory. While HSB staff members have been using WREJ in a production environment without any obvious problems, a significant amount of testing remains to be performed. One of the most notable limitations to utilizing auxiliary BLOB data is that of portability. Only the Geodatabase data model supports the use of the BLOB data type. Conversion of Geodatabase feature classes to shape files results in the loss of all auxiliary BLOB data. However, considering that the Shape file data model will one day be as archaic as the Arc Info coverage model, this is most likely a minor manifestation of evolution. BLOBs are here to stay when it comes to geospatial data (Figure 5).

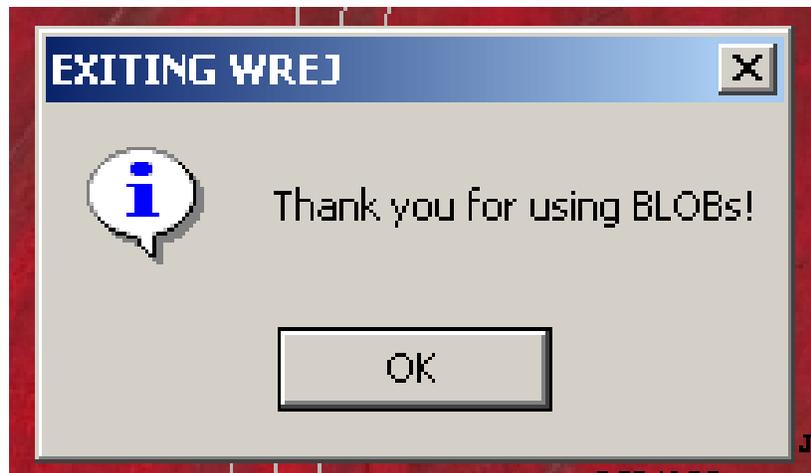


Figure 5: The Beginning of The End for File-based Spatial Data!

AUTHOR INFORMATION:

Stephen N. Hayes
GIS Specialist / ArcSDE DBA
New Mexico Office of the State Engineer
Information Technology Services Bureau, Administrative Services Division
Bataan Memorial Building, Rm. #102
PO Box 25102
Santa Fe, NM 87504-5102
(505) 827-6321
FAX (505) 827-6069
shayes@ose.state.nm.us

Liz Ayarbe
GIS Analyst / Hydrographic Survey Specialist
New Mexico Office of the State Engineer
Hydrographic Survey Bureau, Litigation and Adjudication Program
130 S. Capitol Pl.
PO Box 25102
Santa Fe, NM 87504-5102
(505) 827-4264
FAX (505) 827-7874
layarbe@ose.state.nm.us

Christina Noftsker
GIS Analyst / Hydrographic Survey Specialist
New Mexico Office of the State Engineer
Hydrographic Survey Bureau, Litigation and Adjudication Program
130 S. Capitol Pl.
PO Box 25102
Santa Fe, NM 87504-5102
(505) 827-7849
FAX (505) 827-7874
cnoftsker@ose.state.nm.us