Appendix A: Description of Sediment Load Estimation Model

A geographic information systems (GIS) model was used to estimate sediment loads to streams and approximate the model used in the 2001 sediment total maximum daily loads (TMDLs) for the North Fork Coeur d'Alene River Subbasin. This appendix describes the model in detail and Appendix B contains the code used for the model. Python, a programming language, was used to automate a collection of common GIS-related overlay and data management techniques to make the model strictly repeatable. The script was originally written for ArcGIS version 9.2 and Python version 2.4 and was successfully tested with ArcGIS version 9.3.1 and Python version 2.5.

There are three instances where the script will fail to complete. Script operations are heavily dependent on roads in a watershed and all three instances of failure are road-related. The first case happens when there are no roads within a watershed. In this case, land use becomes the only variable and the model may easily be completed manually by adding up the land use contribution to sediment delivery. The second case happens when there are no road intersections. The third case occurs when all intersections of roads are perfectly straight i.e., when overlapping end-nodes (an intersection) have an angle of 180° relative to one another. Cases where the second and third script failures occur are very rare. None of the script failures was encountered during the course of modeling the nineteen watersheds examined.

Preparation of GIS layers

There are four classes of shapefile required to run the TMDL script: watersheds to be modeled, streams, roads, and land use. All layers must be in a metric Universal Transverse Mercator (UTM) projection. Multiple watersheds and multiple model years are allowed as inputs for a single run.

Streams:

Only one stream layer can be input to the model in a single run. It must be a polyline shapefile and encompass all watersheds to be modeled. The script currently specifies that the streams layer must be named "cda_streams.shp." The model's script will not take into account any information contained in its attribute table. This is the simplest layer to prepare. DEQ uses the 1:100,000 national hydrography dataset (NHD) from the U.S. Geological Survey (USGS) as the primary stream layer for surface water management programs.

Watersheds:

Watershed layers must be polygon shapefiles containing only a single record. The attribute table must contain a field (text data type) called "NAME" for the name of the watershed. No other fields in the attribute table are required and may be deleted. Observe a strict naming scheme for watershed layers; watershed layer names must begin with "wshd" followed by the name of the watershed, e.g. "wshd_clinton.shp" or "wshd_picnic.shp". Multiple watersheds may run through the script so long as this naming scheme is strictly adhered to. The model may be used for watersheds at any scale. Standard watershed layers from USGS and Idaho Department of Water Resources (IDWR) are recommended.

ROADS:

Road layers must be polyline shapefiles and specific to a model year. All road features decommissioned up to and including that specific model year must be deleted. For example, for model year 2007, roads that were decommissioned in 2007 and prior were deleted from the shapefile. The attribute table for road layers must contain a field (float data type) called “CWE” for the CWE road erosion score for roads. “CWE” is the only field in the attribute table required and cannot contain null values. Observe a strict naming scheme for road layers; road layer names must begin with “road” followed by the four-digit model year, e.g. “road_1986.shp” or “road_2007.shp.” Multiple model years may be run provided this naming scheme is strictly adhered to and land use layers for the same model years are provided. For this study, road layers were obtained from the U.S. Forest Service (USFS) Coeur d’Alene River Ranger District.

LAND USE:

Land use layers were developed from harvest information originally provided in a spreadsheet from the USFS Coeur d’Alene River Ranger District. For this study, all watersheds were comprised of a forested land use and sediment delivery coefficients depended on harvest history. The most important data were timber stand identification numbers, harvest types, number of acres harvested, and size of timber stands in acres. This information was used to calculate the sediment delivery coefficients, weighted by area, for every timber stand with a history of harvest within ten years of the model year. This information was calculated in a spreadsheet and then joined to the attribute table of the timber stand layer, a polygon shapefile, using the timber stand identification number as the basis of the join. The result of the join operation resulted in many timber stands lacking a sediment delivery coefficient. This happened when there was no history of harvest within ten years of the model year. The natural background sediment delivery coefficient was applied to those stands as well as stands with no history of harvest at all. The field in the attribute table holding sediment delivery coefficients must be a float data type called “SEDDELCOEF”. Another necessary field in the timber stand’s attribute table is “MODEL_YEAR.” Calculate the field “MODEL_YEAR” for every record as the targeted model year. Observe a strict naming scheme for stand layers; stand layer names must begin with “stnd_” followed by the four-digit model year, e.g. “stnd_1986.shp” and “stnd_2007.shp”. Multiple model years may be run may be run provided this naming scheme is strictly adhered to and road layers built for the same model years are provided.

RUNNING THE MODEL

The model script will allow multiple watersheds and multiple model years to be run. For example, 19 watersheds were modeled for this study. Each of those 19 watersheds was modeled twice, once for model year 1986 and once for model year 2007. This effort required preparation of 24 shapefiles: one stream layer, 19 watershed layers, two road layers, and two stand layers.

All prepared GIS layers must be placed into a single folder. The script for this example specifies that this folder must be C:\GIS\TMDL\TMDL_MODEL. An output folder called "output" should also be provided (C:\GIS\TMDL\TMDL_MODEL\output). When these conditions are met, double-click the script file "NFCDA_TMDL.py" in Windows Explorer to start it. The script may also be run from a Python editor such as PythonWin. The script will take several minutes to complete depending on the number of years and watersheds modeled. For this study, the script took approximately 45 minutes for completion.

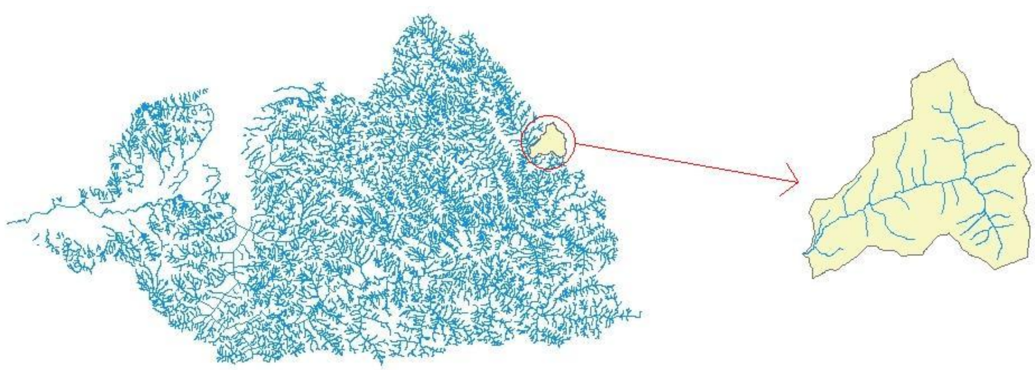
HOW THE TMDL MODEL SCRIPT WORKS

1. The first major step the script automates is to create a list of watershed layers in memory and select the first watershed layer to begin work on. The four lines of Python code that does this are:

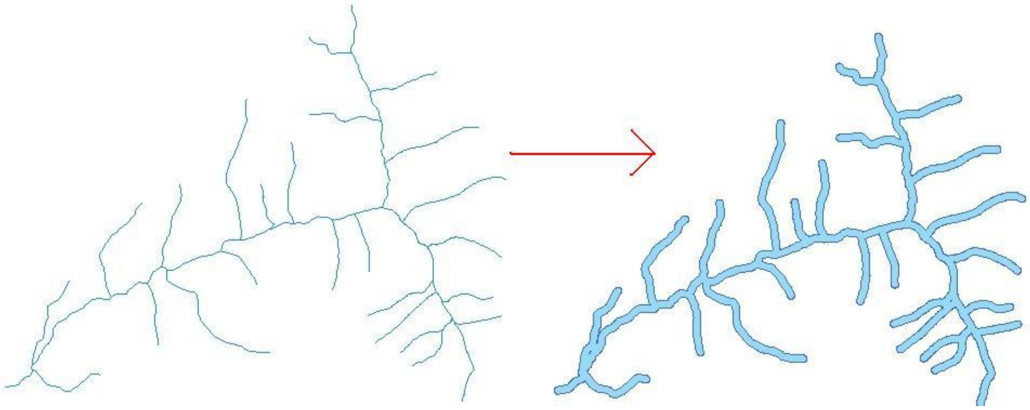
```
wshds = gp.listfeatureclasses("wshd*")
wshds.reset()
wshd = wshds.next()
while wshd:
```

   The first line creates an alphanumeric list with feature classes having names that begin with "wshd." This is why it is important to name the watershed files properly during their preparation. The fourth line creates a loop; a number of operations will be performed on the selected watershed, then the next watershed is selected and the loop repeats until there are no more watersheds in the list.

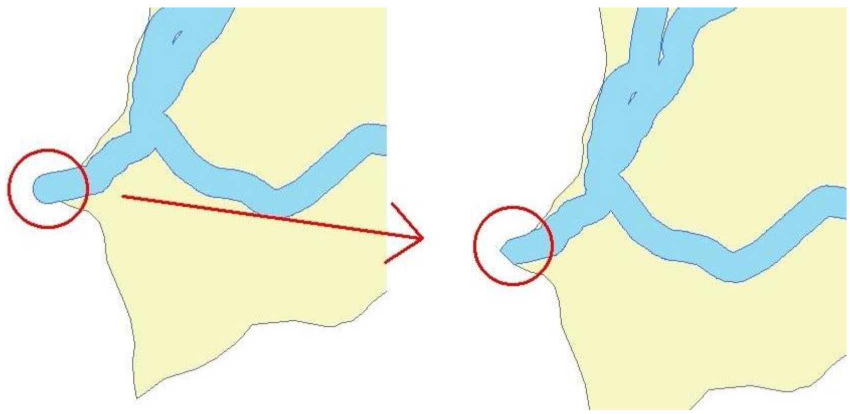
2. A working stream layer is created by clipping the streams layer to the extent of the selected watershed.



3. Next, a 200-foot buffer is created around the streams within the selected watershed. The buffered area is used to identify sections of roads within 200 ft of the stream.

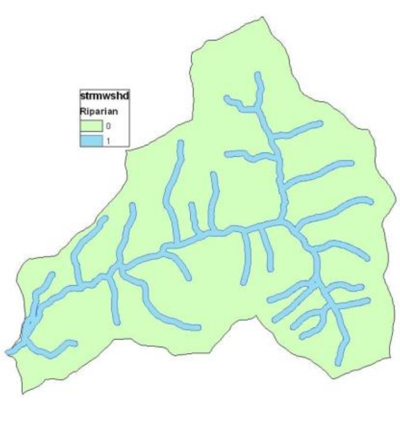


4. As depicted below, some of the buffered stream layer may then fall outside of the watershed boundary during the buffer operation. This occurs most frequently at the mouths of streams. The buffered stream layer is clipped to the watershed to clean this up this artifact.



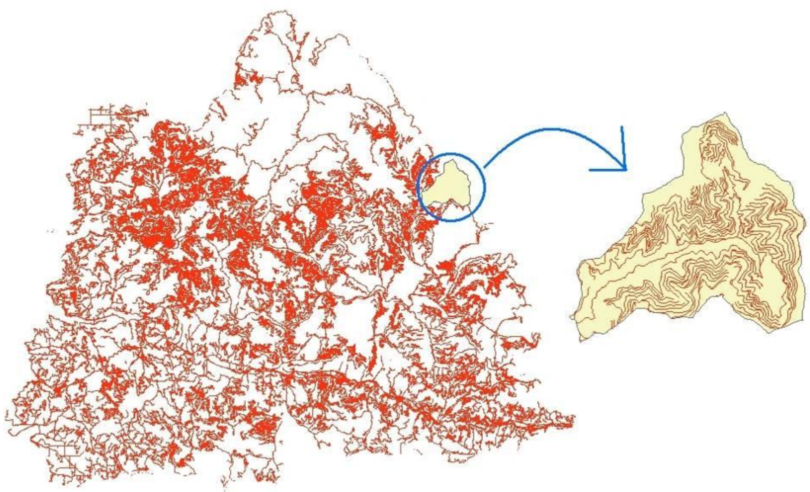
5. A field called “Riparian” is added to the attribute table of the buffered stream layer. This is a binary field denoting whether the area is within the 200 foot riparian zone or not. All records are set to “1” in the riparian field.

6. The buffered stream layer is unioned to the watershed layer. The result in the attribute table with respect to the Riparian field is a “1” in polygons within 200 feet of a stream and “0” outside the riparian corridor.



7. The next step creates two lists of feature classes in memory. The first list is composed of the prepared timber stand layers. This list is created by searching for filenames that begin with "stnd." The second list is composed of the prepared road layers, file names that begin with "road." Both of these lists are in alphanumeric order, a property which helps coordinate the model years between the stand and road layers. An inner loop is set to run the first model year for both the stand and road layers. After a number of operations are run, the next model year for road and stands are selected and the loop repeats until there are no more model years in the lists.

8. A working road layer is created by clipping the selected roads layer to the extent of the selected watershed.



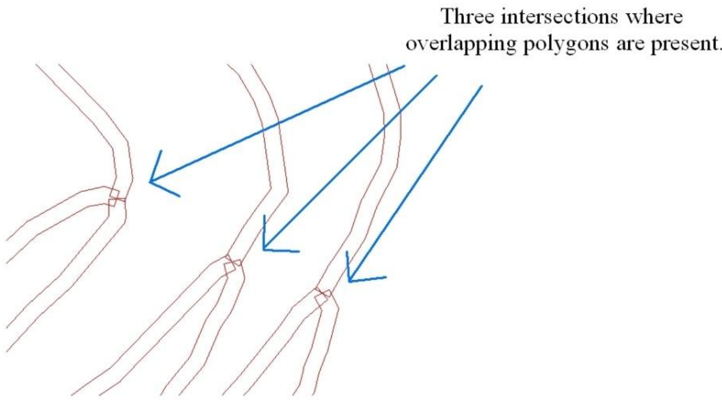
9. A sediment delivery equation is applied to each feature in the clipped road layer. This results in a new field and attribute table containing a sediment delivery coefficient with units in tons/acre/year.

10. Roads within the watershed are buffered 20 ft on each side to account for an assumed average 40 ft wide road prism.

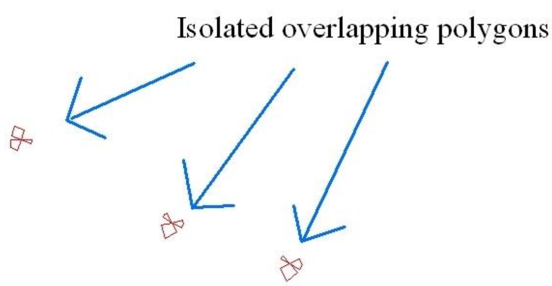
11. Some of the buffered road layer may have fallen outside of the watershed boundary during the buffer operation; this occurs where roads exit the watershed boundary. The buffered road layer is clipped to the watershed to clean up these artifacts.

12. The road buffer operation creates additional unwanted artifacts at or near road intersections. Since each feature is buffered independently, overlapping polygons near road intersections are

created. Removing these artifacts can be achieved in six steps as described below.



12a. First, the overlapping polygons need to be isolated. The buffered road layer is run through the intersect operation by itself. Overlapping polygons are only isolated at this point; they are still stacked. All information in the attribute table is retained, the most important being the sediment delivery coefficient.



12b. The isolated overlapped sections are dissolved. This eliminates stacking and deletes records of overlapped polygons, but at the same time all fields in the attribute table are lost. Simplified, unstacked polygons remain. There is no visual difference between the figure in step 12a and the output of 12b. There is one important field gained from this operation; each dissolved polygon receives a unique feature ID (FID) field which will be carried through the attribute tables of forthcoming map layers.

12c. The outputs of steps 12a and 12b are intersected. This operation retrieves the attribute information lost from the dissolve in step 12b but the resultant map layer again has stacked polygons that must be eliminated.

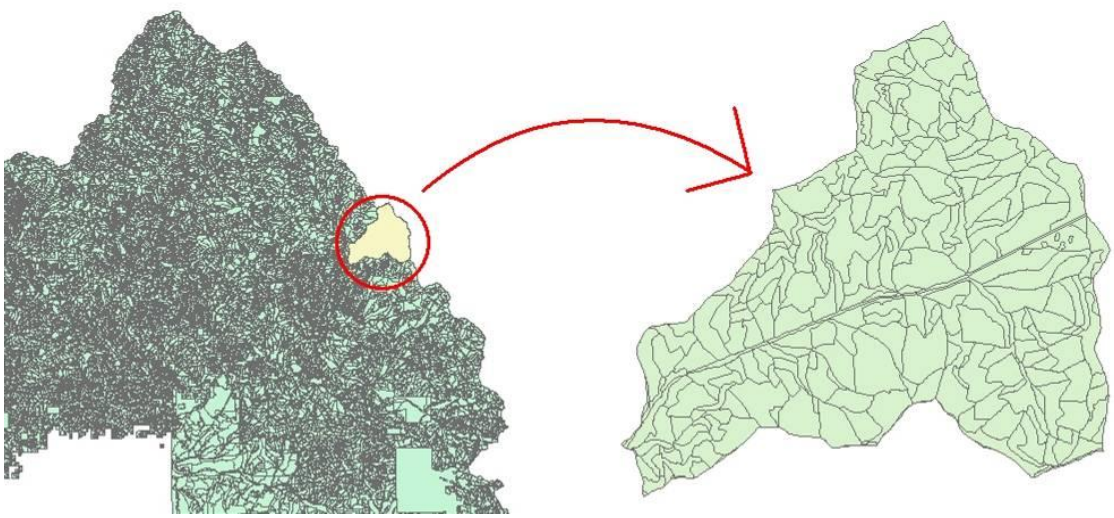
12d. The new overlapping polygon layer is dissolved again to eliminate stacked features. This dissolve operation is run with two options that will retain the sediment delivery coefficient in a summarized form. First, the data is aggregated around the FID field produced by the previous dissolve. Secondly, the polygon with the maximum sediment delivery coefficient is retained; a new field is automatically created for this maximum sediment delivery coefficient.

12e. The areas where there were overlapping polygons in the original buffered road layer (produced at step 11) are erased and the isolated, treated polygons (produced at step 12d) are unioned back in.

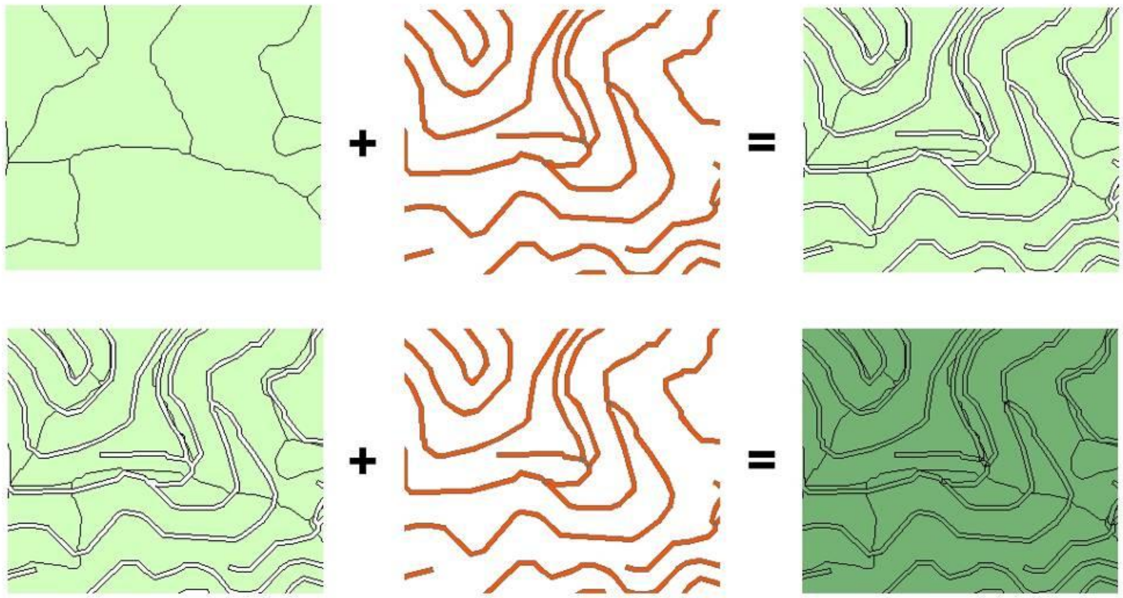
12f. The final step in the treatment of overlapping road polygons is minor data management in the attribute table. Sediment delivery coefficients in polygons that were formerly stacked are contained in a different field than that of road polygons that were never overlapped. These fields are reconciled into a single field.

13. Now the estimated sediment delivery rate is calculated for each section of road. The area of each road polygon is multiplied by its delivery coefficient. Roads that lie within the riparian corridor are allocated 100% of the estimated sediment production while non-riparian roads are allocated 10% of estimated sediment production.

14. A working land use layer is created by clipping the selected timber stand layer to the extent of the selected watershed.

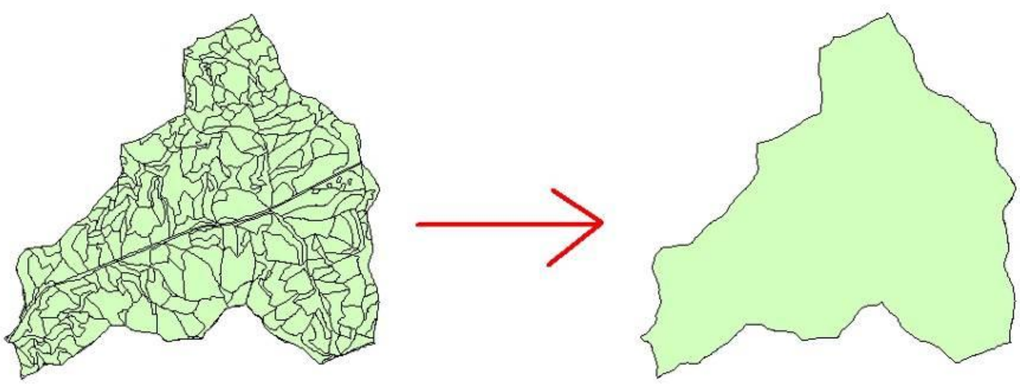


15. The area occupied by roads is erased from the stands layer, and then the roads are unioned back in to the erased area.



16. Delivery coefficients from polygons not associated with roads are multiplied by their area to give the sediment delivery rate in units of tons/acre/year.

17. The final map layer is an integration of timber stands, riparian and non-riparian roads for the selected watershed. This layer is dissolved with the option to sum the sediment delivery. The sum of the sediment delivery is the sole record in the attribute table. This final output is copied to the output folder.



18. The inner loop set up by step 7 is now finished. If there are more model years to run, then the script will loop back to step 8. If no more model years are present, then the script proceeds to step 19.

19. The watershed loop set up by step 1 is now finished. If there are more watersheds to run, then the script will loop back to step 2. If no more model years are present, then the script is finished and will exit.

20. The output folder now contains dissolved watershed polygon shapefiles for every model year and watershed fed into the model. These shapefiles have only one record each, with a field that containing the sum total of sediment delivery rates from riparian roads, non-riparian roads, and the rest of the land. These figures can be opened in ArcMap or ArcCatalog and retrieved.

Appendix B: Sediment Load Estimation Model Script

```
############################################################
##  Total Maximum Daily Load Model
##  North Fork Coeur d'Alene River
##
##  By: Dave Funk
##      Fisheries/Hydrology Technician
##      US Forest Service
##      Idaho Panhandle National Forests
##      Coeur d'Alene River Ranger District
##
##  This project was implemented in cooperation with
##  Idaho Department of Environmental Quality through
##  contacts with Kajsa Stromberg and Tyson Clyne.
##
##  Acknowledgments:
##      Ed Lider - USFS
##      John Ruebke - USFS
##      Jackie McGillivray - USFS
##      Matt Davis – USFS
##
############################################################

print "script running"

# Import system modules.
import arcgisscripting

# Create the Geoprocessor object.
gp = arcgisscripting.create()

# Set workspace.
gp.workspace = "C:\\GIS\\TMDL\\TMDL_MODEL"

# Local variables.
# These are intermediate feature classes and will be deleted later.
strm0 = "strm.shp"
wshd0 = "wshd.shp"
road0 = "rd.shp"
road1 = "rd_clip.shp"
road2 = "rd_buf.shp"
road3 = "rd_buf_clip.shp"
road4 = "rd_buf_intsct1.shp"
road5 = "rd_buf_disolv1.shp"
road6 = "rd_buf_intsct2.shp"
road7 = "rd_buf_disolv2.shp"
road8 = "rd_buf_erase.shp"
road9 = "prepped_road.shp"
strm1 = "strm_clip.shp"
strm2 = "strm_buf.shp"
strm3 = "prepped_strm.shp"
sw = "strmwshd.shp"
```

```
swr = "strmwshdroad.shp"
stnd0 = "stnd.shp"
stnd1 = "stnd_clip.shp"
stnd2 = "stnd_erase.shp"
swrs = "strmwshdrdstnd.shp"
swrsdiss = "swrsdiss.shp"

# There is only one stream layer.
# Everything else must be looped through.
strm = "cda_strms.shp"

# Copy stream feature class so as not to modify the original or be interupted
# by shared schema locks that may exist from other application
# (ArcMap, ArcCatalog, etc.)
gp.copy_management(strm, strm0)

# Get list of watershed feature classes to be modeled and loop through them.
wshds = gp.listfeatureclasses("wshd*")
wshds.reset()
wshd = wshds.next()
while wshd:

    # Copy watershed feature class so as not to modify the original or be interupted
    # by schema locks that may exist from other application (ArcMap, ArcCatalog, etc.)
    gp.copy_management(wshd, wshd0)

    # Clip the stream layer to the watershed to be analyzed.
    gp.Clip_analysis(strm0, wshd0, strm1)

    # Buffer the stream layer 200 feet.
    # We'll find out what roads are within this buffer later.
    gp.Buffer_analysis(strm1, strm2, "200 Feet", "FULL", "ROUND", "ALL", "")

    # Object strm1 is no longer needed.
    gp.delete_management(strm1)

    # Clip the stream to the watershed again.
    # Some of the buffer may have fallen outside of the delineated watershed.
    gp.Clip_analysis(strm2, wshd0, strm3)

    # Object strm2 is no longer needed.
    gp.delete_management(strm2)

    # A binary field denoting riparian polygon.
    # 1 = in riparian and 0 = out of riparian.
    # The field will be carried through overlays with the road polygon later.
    # Roads will be labeled '0' or '1' in the 'Riparian' field.
    gp.AddField_management(strm3, "Riparian", "SHORT", "1", "", "", "", \
    "NON_NULLABLE", "NON_REQUIRED", "")
    gp.CalculateField_management(strm3, "Riparian", "1", "VB", "")

    # Union watershed and stream.
    # Result has polygons of non-riparian or riparian (0 or 1)
    gp.Union_analysis(strm3 + ";" + wshd0, sw, "ALL", "", "GAPS")
```

```
# Stand layer preparation requires calculation of the
# sediment delivery coefficient prior to running this script.
# Get a list of timber stand feature classes and loop through
#them concurrently with road.
stnds = gp.listfeatureclasses("stnd*")
stnds.reset()
stnd = stnds.next()

# Get list of road feature classes for each model year and loop through them.
roads = gp.listfeatureclasses("road*")
roads.reset()
road = roads.next()
while road:

    # Copy road feature class so as to not modify the originals
    #or be interupted by shared schema locks that may exist from
    #other applications (ArcMap, ArcCatalog, etc.).
    gp.copy_management(road, road0)

    # Clip the large road layer to the watershed to be analyzed.
    gp.Clip_analysis(road0, wshd0, road1)

    # Add a field for the sediment delivery coefficient.
    gp.AddField_management(road1, "a_coeff", "DOUBLE", "15", "5", "", "", \
    "NON_NULLABLE","NON_REQUIRED", "")

    # Populate the field for the sediment delivery coefficient.
    # It is the result of the McGreer equation divided by
    # the number of acres in a polygon one mile long and 40 feet wide.
    # Dimensions of the result are tons/acre/year.
    urows = gp.updatecursor(road1)
    urows.reset()
    urow = urows.next()
    while urow:
        cwe = urow.getvalue("CWE")
        acoeff = ((0.0005 * pow(cwe,3)) - (0.0136 * pow(cwe,2)) + \
        (0.3089 * cwe)) / 4.848485
        urow.setvalue("a_coeff", acoeff)
        urows.updaterow(urow)
        urow = urows.next()

    # Release schema locks from the UpdateCursor object.
    del acoeff, cwe, urow, urows

    # Buffer the roads to account for an assumed 40 foot wide road prism.
    # Note that there is no dissolve to preserve the CWE field.
    # Will result in polygons that overlap at intersections.
    # Overlapping polygons will be fixed in the next few steps.
    gp.Buffer_analysis(road1, road2, "20 Feet", "FULL", "FLAT", "NONE", "")

    # Clip the buffered roads to make sure the polygons didn't
    #spill outside the watershed.
    gp.Clip_analysis(road2, wshd0, road3)

    # Intersect the clipped, buffered road with nothing.
```

```
# Isolates the overlapping sections of the buffered polygons.
gp.Intersect_analysis(road3, road4, "ALL", "", "INPUT")

# Dissolve the overlapping sections of the buffered polygons.
# Removes the overlap, but all informational fields deleted.
# Note Multipart Features not allowed - unique FID field created
# for every overlapped area.
gp.Dissolve_management(road4, road5, "", "", "SINGLE_PART")

# Intersect to retrieve informational fields.
# Overlap has returned, but now have a unique field to dissolve on.
# FID from previous dissolve layer gets renamed to 'FID_rd_b_2'.
gp.Intersect_analysis(road4 + ";" + road5, road6, "ALL", "", "INPUT")

# Dissolve on field previously mentioned.
# In order to maintain the conservative nature of the model,
# maximum sediment delivery coefficients of the overlapping polygons
# are carried into the new, dissolved polygons.
gp.Dissolve_management(road6, road7, "FID_rd_b_2", \
"CWE MAX;a_coeff MAX", "SINGLE_PART")

# Erase areas where there were overlapping polygons from the
#original clipped, buffered road.
gp.Erase_analysis(road3, road5, road8, "")

# Union the new non-overlapping polygons.
gp.Union_analysis(road7 + ";" + road8, road9, "ALL", "", "GAPS")

# Reconcile sediment delivery coefficient fields.
gp.AddField_management(road9, "acoeff", "DOUBLE", "10", "5", \
"", "", "NON_NULLABLE", "NON_REQUIRED", "")
gp.CalculateField_management(road9, "acoeff", "[MAX_a_coef] + \
[a_coeff]", "VB", "")

# Reconcile CWE fields.
urows = gp.updatecursor(road9)
urows.reset()
urow = urows.next()
while urow:
    oldcwe1 = urow.getvalue("MAX_CWE")
    oldcwe2 = urow.getvalue("CWE")
    newcwe = oldcwe1 + oldcwe2
    urow.setvalue("CWE", newcwe)
    urows.updaterow(urow)
    urow = urows.next()

# Release schema locks from the UpdateCursor object.
del newcwe, oldcwe2, oldcwe1, urow, urows

# Add and populate a field for road sediment delivery.
# Note that this is still 100% efficiency.
# 10% efficiency will be applied to non-riparian roads later.
gp.AddField_management(road9, "rd_sed_del", "DOUBLE", "15", \
"4", "", "", "NON_NULLABLE","NON_REQUIRED", "")
urows = gp.updatecursor(road9)
```

```
urows.reset()
urow = urows.next()
while urow:
    sqm  = urow.getvalue("Shape").Area
    acres = sqm / 4046.825
    acoeff = urow.getvalue("acoeff")
    rdseddel = acoeff * acres
    urow.setvalue("rd_sed_del", rdseddel)
    urows.updaterow(urow)
    urow = urows.next()

# Release schema locks from the UpdateCursor object.
del rdseddel, acoeff, acres, sqm, urow, urows

# Clean up attribute table by deleting fields containing old temporary data.
gp.DeleteField_management(road9, \
"FID_rd_buf;FID_rd_b_2;MAX_CWE;MAX_a_coef;FID_rd_b_1;BUFF_DIST;a_coeff")

# Intersect prepped roads into previous output.
# Result is buffered roads now known to be riparian or non.
gp.Intersect_analysis(road9 + ";" + sw, swr, "ALL", "", "INPUT")

# Objects road[0-9], are no longer needed.
gp.delete_management(road0)
gp.delete_management(road1)
gp.delete_management(road2)
gp.delete_management(road3)
gp.delete_management(road4)
gp.delete_management(road5)
gp.delete_management(road6)
gp.delete_management(road7)
gp.delete_management(road8)
gp.delete_management(road9)

# Add and populate field for actual road sediment
#delivery given riparian efficiency coefficient.
gp.AddField_management(swr, "sed_del", "DOUBLE", "10", "5", \
"", "", "NON_NULLABLE", "NON_REQUIRED", "")

# Riparian roads given 100% delivery efficiency.
urows = gp.updatecursor(swr, "Riparian = 1")
urows.reset()
urow = urows.next()
while urow:
    seddel = urow.getvalue("rd_sed_del")
    urow.setvalue("sed_del", seddel)
    urows.updaterow(urow)
    urow = urows.next()

# Release schema locks from the UpdateCursor object.
del seddel, urow, urows

# Non-Riparian roads given 10% delivery efficiency.
urows = gp.updatecursor(swr, "Riparian = 0")
urows.reset()
```

```
urow = urows.next()
while urow:
    xseddel = urow.getvalue("rd_sed_del")
    seddel = xseddel * 0.1
    urow.setvalue("sed_del", seddel)
    urows.updaterow(urow)
    urow = urows.next()

# Release schema locks from the UpdateCursor object.
del seddel, xseddel, urow, urows

# Delete unneeded fields from result.
gp.DeleteField_management(swr, \
"FID_preppe;rd_sed_del;FID_strmws;FID_prep_1;Id;FID_wshd")

# Copy timber stand feature class so as not to
#modify the original or be interrupted by shared schema
#locks that may exist from other application (ArcMap, ArcCatalog, etc.)
gp.copy_management(stnd, stnd0)

# Clip the stand feature class to the watershed to be analyzed.
gp.clip_analysis(stnd0, wshd0, stnd1)

# Object stnd0 no longer needed.
gp.delete_management(stnd0)

# Put together a final output name.
fcs = gp.searchcursor(wshd0)
fcs.reset()
fc = fcs.next()
wshdname = string.replace(fc.getvalue("NAME"), " ", "_")
del fc, fcs
fcs = gp.searchcursor(stnd1)
fcs.reset()
fc = fcs.next()
year = fc.getvalue("MODEL_YEAR")
opname = wshdname + "_" + year + ".shp"
del year, wshdname, fc, fcs

# Erase roads from stand, then union to join all features.
gp.erase_analysis(stnd1, swr, stnd2)
gp.union_analysis(swr + ";" + stnd2, swrs, "NO_FID", "", "GAPS")

# Objects stnd[1-2] and swr no longer needed.
gp.delete_management(stnd1)
gp.delete_management(stnd2)
gp.delete_management(swr)

# Reconcile sediment delivery field.
urows = gp.updatecursor(swrs, "sed_del = 0")
urows.reset()
urow = urows.next()
while urow:
    coeff = urow.getvalue("SEDDELCOEF")
    sqm = urow.getvalue("SHAPE").Area
```

```
            acres = sqm / 4046.825
            seddel = coeff * acres
            urow.setvalue("sed_del", seddel)
            urows.updaterow(urow)
            urow = urows.next()

        # Release schema locks from the UpdateCursor object.
        del seddel, acres, sqm, coeff, urow, urows

        # Dissolve all and sum the sediment delivery field.
        gp.Dissolve_management(swrs, swrsdiss, "", "sed_del SUM", "MULTI_PART")

        # Copy to output folder.
        gp.copy_management(swrsdiss, "output\\" + opname)
        print opname + " done"

        # Objects swrs and swrsdiss are no longer needed.
        gp.delete_management(swrs)
        gp.delete_management(swrsdiss)

        # Get next stand and road and loop through the next model year.
        # Note that road is not dedented.
        # This prevents heterogeneous years between the stands and roads overlays
        # so long as the all the input feature layers are named
        #properly as documented.
        stnd = stnds.next()
        road = roads.next()

    # Delete current and get next watershed.
    gp.delete_management(wshd0)
    gp.delete_management(strm3)
    gp.delete_management(sw)
    wshd = wshds.next()

# Delete the temporary stream feature class
gp.delete_management(strm0)

print "script completed"
```

Appendix C. BURP Site Photographs

Figure C-1. Big Elk Creek Site (2009SCDAA001)

a. Bottom of the Big Elk Creek site looking upstream.

b. Top of the Big Elk Creek site looking downstream.

Figure C-2. Cougar Gulch Site (2009SCDAA002)

a. Bottom of the Cougar Gulch site looking upstream.

b. Top of the Cougar Gulch site looking downstream.

Figure C-3. East Fork Steamboat Creek Site (2009SCDAA003)

a. Bottom of the East Fork Steamboat Creek site looking upstream.

b. Top of the East Fork Steamboat Creek site looking downstream.

Figure C-4. Picnic Creek Site (2009SCDAA004)

a. Bottom of the Picnic Creek site looking upstream.

b. Top of the Picnic Creek site looking downstream.

Figure C-5. Upper Tepee Creek Site (2009SCDAA005)

a. Bottom of the Upper Tepee Creek site looking upstream.

b. Top of the Upper Tepee Creek site looking downstream.

Figure C-6. Yellowdog Creek Site (2009SCDAA006)

a. Bottom of the Yellowdog Creek site looking upstream.

b. Bottom of the Yellowdog Creek site looking downstream.

Figure C-7. Skookum Creek Site (2009SCDAA007)

a. Bottom of the Skookum Creek site looking upstream.

b. Top of the Skookum Creek site looking downstream.

Figure C-8. Stewart Creek Site (2009SCDAA008)

a. Bottom of the Stewart Creek site looking upstream.

b. Top of the Stewart Creek site looking downstream.

Appendix D. PIBO Site Photographs

Figure D-1. Big Elk Creek Site (#2851)

c. Bottom of the Big Elk Creek site looking upstream.

d. Top of the Big Elk Creek site looking downstream.

Figure D-2. Cougar Gulch Site (#2732)

c. Bottom of the Cougar Gulch site looking upstream.

d. Top of the Cougar Gulch site looking downstream.

Figure D-3. East Fork Steamboat Creek Site (#2199)

c. Bottom of the East Fork Steamboat Creek site looking upstream.

d. Top of the East Fork Steamboat Creek site looking downstream.

Figure D-4. Picnic Creek Site (#2730)

c. Bottom of the Picnic Creek site looking upstream.

d. Top of the Picnic Creek site looking downstream.

Figure D-5. Upper Tepee Creek Site (#2850)

c. Bottom of the Upper Tepee Creek site looking upstream.

d. Top of the Upper Tepee Creek site looking downstream.

Figure D-6. Yellowdog Creek Site (#2852)

c. Bottom of the Yellowdog Creek site looking upstream.

d. Bottom of the Yellowdog Creek site looking downstream.

Figure D-7. Skookum Creek Site (#2731)

c. Bottom of the Skookum Creek site looking upstream.

d. Top of the Skookum Creek site looking downstream.

Figure D-8. Stewart Creek Site (#2853)

c. Bottom of the Stewart Creek site looking upstream.

d. Top of the Stewart Creek site looking downstream.

Appendix E. Phase 1 Sediment Model Results

Table E-1. The sediment TMDL evaluation model was used to compare estimated sediment loads from before and after TMDL implementation.

	Sediment Load (t/yr)			*Natural Background Sediment Load (t/yr)*	*Sediment Load Target at 1.5 Times Background (t/yr)*	*Sediment Load as Factor Over Background*	
Watershed	*1986*	*2007*	*Change*			*1986*	*2007*
Big Elk Cr	438	315	-28%	171	256	2.56	1.85
Brett Cr	146	81	-45%	77	116	1.90	1.04
Cabin Cr	108	95	-12%	57	85	1.89	1.68
Clinton Cr	133	73	-45%	66	100	2.02	1.10
Cougar Gulch	930	461	-50%	285	427	3.26	1.62
Downey Cr	399	209	-48%	140	211	2.85	1.49
East Fork Cougar Gulch	72	34	-53%	28	42	2.57	1.21
East Fork Steamboat Cr	595	342	-43%	162	243	3.67	2.11
Goose Cr	95	64	-33%	47	71	2.02	1.35
Hudlow Cr	202	182	-10%	80	120	2.53	2.28
Lewelling Cr	61	60	-1%	31	46	1.97	1.94
Little Tepee Cr	71	54	-24%	40	60	1.78	1.36
Picnic Cr	203	107	-47%	76	113	2.67	1.41
Skookum Cr	257	221	-14%	91	137	2.82	2.42
Spruce Cr	248	167	-33%	149	223	1.66	1.13
Stewart Cr	239	174	-27%	85	127	2.81	2.06
Upper Tepee Cr	403	336	-17%	207	311	1.95	1.62
West Fork Cougar Gulch	242	104	-57%	60	91	4.03	1.72
Yellowdog Cr	324	183	-43%	114	171	2.84	1.60

Appendix F. 2009 BURP Data Summary

Table F-1. BURP stream habitat data from North Fork Coeur d'Alene River Subbasin sites in 2009.

Data	*Big Elk*	*Cougar*	*EF Steamboat*	*Picnic*	*Tepee*	*Yellowdog*	*Stewart*	*Skookum*
Instream Cover (rating)	9	11	6	13	12	12	14	11
Large Organic Debris (count)	7	36	8	26	7	122	88	85
Percent Fine Sediment (%)	10.8	4.4	7.4	0.7	5.7	0.7	7.6	0.5
Riffle Embeddedness (rating)	19	18	16	11	10	18	17	15
Wolman Size Classes (#)	8	8	9	9	9	9	9	8
Channel Shape (rating)	7	3	4	3	6	2	5	2
Percent Bank Cover (%)	89	51	100	85	75	85	33	20
Percent Canopy Cover (%)	23	61	55	88	34	50	29	26
Disruptive Pressures (rating)	4	9	9	9	8	5	9	7
Zone of Influence (rating)	7	9	9	9	8	6	6	6

Table F-2. BURP stream macroinvertebrate data from North Fork Coeur d'Alene River Subbasin sites in 2009.

Data	*Big Elk*	*Cougar*	*EF Steamboat*	*Picnic*	*Tepee*	*Yellowdog*	*Skookum*	*Stewart*
Total Taxa	33	37	44	47	37	33	39	34
Ephemeroptera Taxa	10	10	11	9	7	7	8	7
Plecoptera Taxa	6	7	7	11	8	7	7	6
Trichoptera Taxa	6	8	9	9	9	10	11	8
Percent Plecoptera	23	16	23	34	14	12	42	33
Hilsenhoff Biotic Index (HBI)	4.97	6.05	5.36	5.33	5.35	5.27	4.80	4.96
Percent 5 Dominant Taxa	57	58	54	61	66	51	68	59
Scraper Taxa	8	9	9	7	10	8	6	7
Clinger Taxa	26	28	30	31	27	26	28	22

Table F-3. Stream fish data from North Fork Coeur d'Alene River Subbasin sites in 2009.

Data	*Big Elk*	*Cougar*	*EF Steamboat*	*Picnic*	*Tepee*	*Yellowdog*	*Skookum*	*Stewart*
Number of Coldwater Native Species	2	1	2	2	2	2	2	2
Percent Coldwater Individuals	100	100	99.6	100	100	95.8	100	100
Percent Sensitive Native Individuals	5.6	0.0	2.5	14.9	43.7	12.5	0.3	10.3
Number of Coldwater Individuals per Minute	7.4	6.8	13.6	13.6	3.3	3.0	5.1	3.8
Number of Sculpin Age Classes	6	6	6	5	6	5	5	4
Number of Salmonid Age Classes	2	3	3	3	2	3	1	2
Presence of Tailed Frogs (T) or Other Native Amphibians (A)	A	T	No	T	T	A	T	T

Appendix G. 2009 PIBO Data Summary

Table G-1. PIBO stream habitat data from North Fork Coeur d'Alene River Subbasin sites in 2009.

Data	*Big Elk*	*Cougar*	*EF Steamboat*	*Picnic*	*Tepee*	*Yellowdog*	*Stewart*	*Skookum*
Large Woody Debris Frequency (#/km)	277	94	117	157	36	171	342	248
Large Woody Debris Volume (m^2)	52	50	84	53	5	105	57	165
Percent Fine Sediment (% <6 mm)	12	16	4	7	8	8	5	8
Percent Undercut Banks (%)	56	24	29	42	40	10	53	32
D50 (mm)	46	66	79	44	39	48	42	65
Bank Angle (degrees)	81	119	107	104	89	140	90	104
Percent Pools (%)	80	47	6	29	71	59	54	10
Residual Pool Depth (m)	0.44	0.64	0.31	0.25	0.31	0.31	0.37	0.29

Appendix H. Water Quality Success Stories