An ESRI® Technical Paper • April 2009

ArcGIS® Server in Practice Series: Large Batch Geocoding

ESRI 380 New York St., Redlands, CA 92373-8100 USA
TEL 909-793-2853 • FAX 909-793-5953 • E-MAIL info@esri.com • WEB www.esri.com

J-9782

ArcGIS Server in Practice Series: Large Batch Geocoding

Contents **Page**

J-9782

ArcGIS Server in Practice Series: Large Batch Geocoding

The ArcGIS® Server in Practice series provides practical information for the configuration and implementation of ArcGIS Server solutions. Documents in this series use well-defined user workflows and system configurations as a storyboard for describing best practices with ArcGIS Server. The series focuses on these workflows from a holistic point of view. It provides practical information that will help you understand how to design and configure ArcGIS Server optimally.

Executive Summary

Places had names long before they had coordinates; naming schemes evolved to distinguish locations with precision and without ambiguity. In a geographic information system (GIS), places have coordinates; **geocoding** is the process of deriving a geographic coordinate from textual input, typically a street address, and may be performed in reverse to find an address from a geographic location. Related functions are the validation of addresses, which tests that an address is physically deliverable, and standardization, which ensures that address data is in standard formats suitable for organizational usage and database hygiene. This document describes an ArcGIS Server Standard Enterprise configuration hosting geocoding services suitable for both a heavy transactional load of interactive geocodes and large-scale batch geocoding. This configuration uses national scale data for the United States, available from ESRI. Similar national scale datasets are available from ESRI for Canada and Europe and from distributors and business partners in other countries around the world.

Business Requirement Scenario

The user is a large retailer and geocodes customer transactions primarily to support analysis of facility location metrics such as travel distance and demographic segmentation studies. Patronage of each store is studied in each market segment (that is, region and demographic profile) to provide input for effective promotional activity, find if store locations under- or over-compete in a geographic area (including with each other), and determine how competitor facilities are impacting sales. In addition to store location and national advertising decisions, the company's loyalty scheme is tuned in to each geographic area with in-store benefits, notifying customers by e-mail about significant events like a store opening, change in local road navigability, product launch in a market, or change in customer behavior. In short, the ability to geocode customer transactions is the company's source of competitive advantage.

Workflow Scenario

In this scenario, there are two main geocoding workflows:

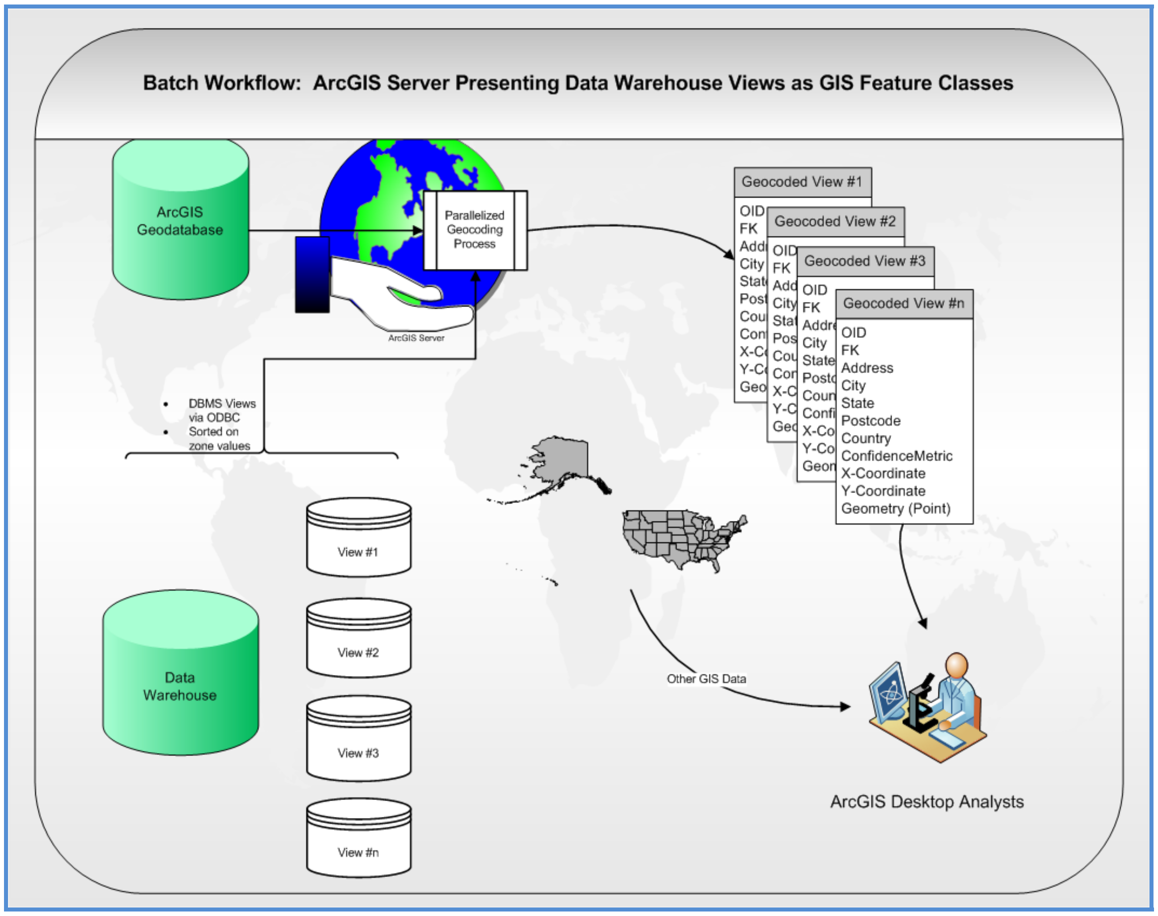
- Continuous transactional geocoding and map request load coming from an intranet Web application used in point-of-sale customer detail capture
- On-demand batch geocoding load for the market analytics department, which uses ArcGIS Desktop and the corporate data warehouse

Of these two, the interactive geocode transactions and associated map image requests to support reverse geocoding are expected to be in the low hundreds of thousands per day, while the batch geocode transactions are an order of magnitude larger (many millions at a time) and, although occasional, must nevertheless complete during a working day or overnight.

The Web application supporting transactional geocodes makes a map request for each record processed so the remote user can validate the address with the customer, and it is expected that about 10 percent of the time an extra map request and reverse geocode will be required in cases where the customer identifies a location. It is evident from published examples of cached map services and geocoding services (http://www.esri.com/software/arcgis/arcgisserver/success_stories.html) that transaction volumes in the hundreds of thousands per day are not high end. Considered together, the interactive geocodes, cached map delivery, and reverse geocodes are not of sufficient volume to be nearly as significant to system design as the massive batch geocoding requirement to support the market analytics department. In any event, should the Web application be found to have predictable peak demand episodes, the batch geocoding processes can be scheduled outside these—provided, of course, the system and processes are designed to complete in the available time window.

It is known that the batch geocoding requirement may be many millions of records per transaction. These batch jobs are consistent in terms of data source (ultimately the data warehouse) and output destination (a feature class in the enterprise geodatabase) and are driven by queries on the company data warehouse engine. Transactions occur for situations such as the update of the geocoding reference data, changes in customer address details, errors being fixed, and new geocoding preferences being implemented. The GIS platform capacity is determined by the most demanding of these batch use cases—namely, the geocoding of the entire customer database on a semiannual basis when the geocoding reference data vendor supplies an update.

Conceptual Solution Diagram—Batch Processing



Design Considerations

Workflow Drives the Design

The company's data warehouse manages many millions of customer records. The data warehouse product is not spatially aware, but it is the system of record where data hygiene is managed, and the company has made a decision to upload geocoded coordinates alongside the existing customer records in the warehouse system. One justification for this is that with store and customer coordinate values in the warehouse, simple Euclidean distance calculations can be performed in the warehouse using simple math and can be included in cubes for reporting purposes, leveraging the GIS results inside the warehouse.

Batch geocoding is a geoprocessing task that should be handled totally on the GIS server as a single transaction—the latency of passing record sets between a remote client platform, script, or application via cursors or the Web tier should be avoided. This contrasts with the interactive geocoding requirement where lightweight representations of tables and features need to be transmitted between client and server for the end result to be derived and used and, of course, map images to be delivered by the cached service.

While ArcGIS Server is often used to handle frequent short transactions, this document will show how it can also be used for large batch jobs. In fact, one of the advantages of ArcGIS Server is that it allows the creation of workflows by which large jobs are split into smaller chunks that can be executed in parallel, hence maximizing the hardware resources available in a system. This document will describe how to leverage this symmetric multiprocessing approach by running geoprocessing tasks asynchronously.

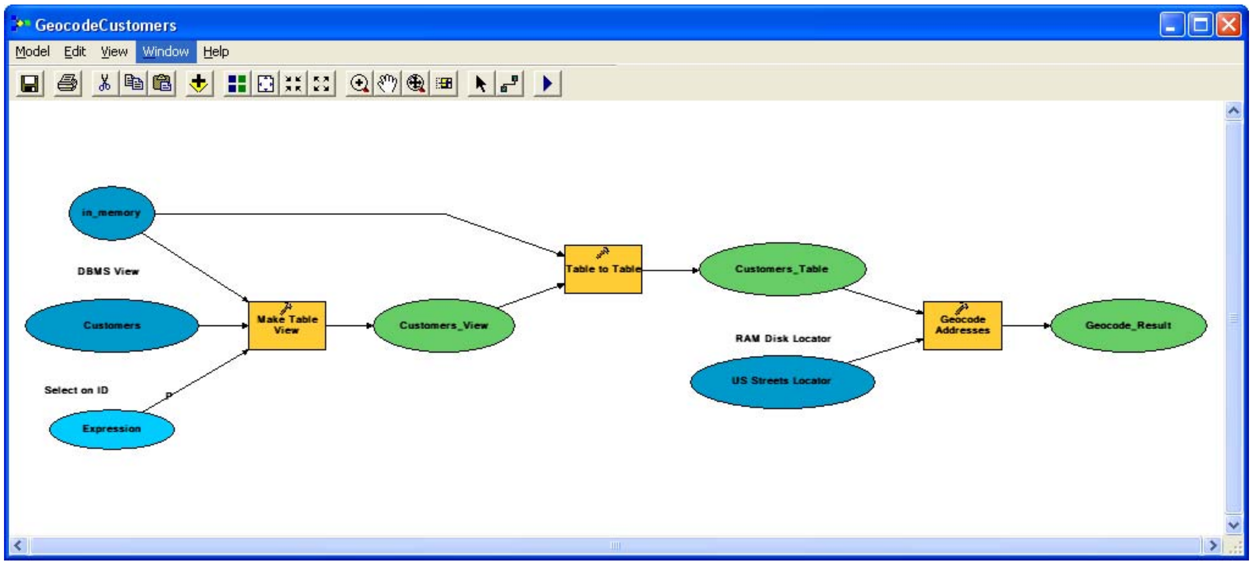
The primary goal is to design a geoprocessing workflow for batch geocoding that optimally uses server resources to meet performance goals and executes entirely on the server.

The approach to achieve this is very simple:

- Implement data warehouse views that can be queried into subsets (by row number).
- Make these views accessible to ArcGIS Server via an OLE DB connection.
- Publish an asynchronous geoprocessing service with a query input parameter that selects from a data warehouse view and creates a geocoded feature class.
- Write a helper script tool that generates multiple data warehouse queries, sends the queries to the geoprocessing service, waits for process completion, and appends the geoprocessing outputs.

The geoprocessing service is straightforward to author; below is a ModelBuilder™ diagram of the service. The Customers variable is an OLE DB connection to a data warehouse view. The Expression input parameter is a SQL statement supplied by the helper script tool (discussed later). The Make Query Table and Table to Table processes have their current workspace set to in_memory for speed (disk I/O is expensive); note that the usual practice of setting the geocoded feature class to an output parameter is not followed (because the data is not passed)—the helper script will pick up the data directly from the job directories.

ModelBuilder Diagram of the Published Geoprocessing Service



At this point, it is worth discussing the actual geocoding process options in detail, as that is where the heaviest processing occurs. Note: All tips, except for the first one about sorting data, will benefit interactive geocoding as well as batch geocoding.

Understanding Geocoding Processing Options

Geocoding Tips

Locators are built using templates provided with ArcGIS, or custom templates designed to meet specific requirements. Locator templates may be tuned in many ways by editing parameters in the locator template (.lot) file before creating the locator or, in some cases, *after* creating the locator by editing the .loc file. Locators are populated with reference data from a chosen vendor. In many cases, the default template properties and reference data from ESRI's data partners are reasonable choices for locators, and a good choice might be to purchase a prebuilt locator. But what are the decision factors to achieve peak performance? Following are some tips in approximate order of importance.

Tip 1

Sorting input data is by far the most important batch geocoding preprocessing step. When using any locator that uses zone fields, and you expect your input data to have many zone values (that is, to access the locator index structure at many places), sorting your data according to the zone field storage in the locator index will achieve significant performance gains. Batch geocoding throughput can easily improve by a factor of 5 when the input table is presorted.

How do you determine the sort order you need and perform the sorting of input tables? The first step is to look at the sort order for your zone fields. This is exposed in the .lot file in the lines for the SortBy parameter. Here is an example snippet from a .lot file:

```
RD.Lox1.SortBy = $IdxTable1.STABBR
RD.Lox1.SortBy = $Table1.State
RD.Lox1.SortBy = $IdxTable1.ZIPL1,$IdxTable1.ZIPR1
RD.Lox1.SortBy = $IdxTable1.ZIPL,$IdxTable1.ZIPR
RD.Lox1.SortBy = $Table1.LeftZIP,$Table1.RightZIP
RD.Lox1.SortBy = $IdxTable1.CTSXL,$IdxTable1.CTSXR
RD.Lox1.SortBy = $Table1.LeftCity,$Table1.RightCity
```

These lines in the .lot file specify that the locator index is sorted on state, left and right ZIP Code™, then left and right city name. The sort order is read from the top down and in a spatial context from coarse to fine zones.

How you sort an input address table on zone fields will depend on the storage workspace. The download available with this document includes a script tool that sorts geodatabase or dBASE tables, creating a new output table. If you are reading from a DBMS (for example, via OLE DB) and your table is not registered with the geodatabase, you can create a view that uses the Order by clause. As in the example above, the view needs to be read using the Make Query Table geoprocessing tool to supply an ObjectID (OID) field.

You may know that locators have a batch presort parameter available. This is pertinent to unsorted data only, but setting this parameter is recommended anyway, to catch the situation where you have not sorted your input data. For a locator with the zone fields being discussed, the locator parameters would be as follows:

BatchPresortInputs = State
BatchPresortInputs = ZIP
BatchPresortInputs = City

The batch presort parameter will not achieve the performance gain experienced when actually sorting your data.

Tip 2 If possible, use a RAM or solid state disk to store your locator(s). Locators are heavily accessed indexes that greatly benefit from being kept in memory (the above tip effectively keeps locator parts in memory between records too). It is easy to configure a RAM disk to hold a locator of known size. A RAM disk has the property that it is volatile (if your computer reboots, you lose your data), but in the case of server class machines with high uptime, this issue is easily managed. You can easily get a 50 percent performance gain in geocoding by using a RAM disk.

Tip 3 If you cannot spare the RAM to implement a RAM disk, or buy a solid state disk, use a file geodatabase workspace for your locators and compact it after creating any locator. Locators are large files that are frequently accessed during geocoding operations, so fragmentation can affect performance. File-based workspaces (directories and file or personal geodatabases) are recommended for locator storage over database workspaces (personal, workgroup, and ArcSDE® workspaces); file and personal geodatabases have the advantage that they can be compacted (defragmented) with ArcGIS and not the operating system.

Tip 4 Use a fast disk array for your geoprocessing job directories. SAN or RAID setups help with the simultaneous writes expected when running multiple asynchronous geocoding processes in the geoprocessing job directories; geocoding is disk intensive.

Tip 5 Understand your requirement for geocoding precision. Unless you really need to locate exactly on a house or building, locator styles that use point ("rooftop") reference data are not indicated. Locators using street centerline data with left and right house number ranges are sufficient in most cases, perhaps with a backup locator that locates to ZIP or postal code.

Tip 6 Decide if you need a locator that uses alternate name queries. ESRI supplies locator templates for reference data with alternate names in row; these are good for exhaustive processing of candidates but have a speed penalty. A U.S. streets locator using alternate names might return 1.5 million geocodes per hour but 2.5 million geocodes per hour for the same input addresses when a style that does not process alternate names is used. You can run an extract, transform, and load (ETL) process to derive a reference dataset that appends extra records having alternate name values into the primary names and achieve the benefits of speed *and* alternate name testing.

Tip 7 Understand your requirement for a composite locator. Composite locators attempt geocodes against a list of locators in an order you design, and access each locator until a match is found or the list is exhausted. They give the best matching outcome but at the expense of speed. If speed is of the essence, then design a geoprocessing workflow that uses single locators and reduces sets of unmatched records in sequence.

Tip 8 Use an appropriate locator when handling nationwide data. It might seem that using any locator that accepts a base address plus ZIP or postal code will suffice, because you know that the zoning code is unique nationwide and resolves any ambiguity in candidate geocodes. This is true, but only by processing a much longer list of candidates per record. Zone fields are processed using a type of soundex comparison, and this comparison is much more efficient at distinguishing candidates when using a combination of zone fields. For nationwide data, use a locator that accepts city and state as well as ZIP Code.

At this point, the script tool used to drive the geoprocessing service is examined. The tool (supplied in the download for this technical paper) is unusual for an ArcGIS Server implementation in that it runs on the server itself. This is because it accesses the server file system—the geoprocessing jobs directory—to pick up job results. There is one input parameter—the part count. This is the number of symmetrical parts that you wish the input data to be divided into and geocoded in parallel. The tool's script handles checking the row count of the input DBMS view, creating the job queries based on row ID, launching the jobs, waiting for the outputs, and appending the results. See appendix 2 for the script source code and example messages returned.

Testing shows that there is no benefit in running either fewer or more parallel tasks than you have CPU cores, so the recommended approach is to use a default part count equal to your available CPU cores. In the test environment at ESRI, an eight-core server was used, so the part count used was 8. **The effective batch geocode performance was 15 million records per hour to retrieve, geocode, and append into a final output feature class.**

Appendix 1

Description of the ArcGIS Server Test Platform

Processor Cores	2x Quad Core 64-bit 3 GHz Intel® Xeon® E5450
Memory	16 GB
Disk Controller	Dell® SAS 6/iR SCSI/RAID Controller
Disk Drives	Dell 136 GB SCSI Disk—Mirrored
RAM Disk	Cenatek RAMDiskSE 4 GB
Operating System	Microsoft® Windows Server® 2003 R2

ArcGIS Server Service Specifics

The geoprocessing model was published as an asynchronous pooled service with a minimum of 1 and a maximum of 16 instances. Users implementing a similar approach should carefully design the maximum time a client can use a service to certainly cover the time the helper script takes to execute such as 24 hours (= 86,400 seconds).

Appendix 2

The Python® source for the helper script tool is followed by an example message stream. This script is also available in the download for this technical paper.

```
# Author: ESRI (#5588)
# Date: February 5, 2009
# Purpose: This script demonstrates a methodology for very
large batch geocoding.
# The scenario is the source table in a DBMS view in a
remote DBMS accessed
# via an OLE DB connection, and the destination feature
class is truncated
# at each run. We are maintaining a spatial edition of the
DBMS table.
#
#
try:
 import arcgisscripting, os.path, time, sys, traceback
 gp = arcgisscripting.create(9.3)
 gp.overwriteoutput = 1

 #Record the start time
 startTime = time.clock()
 gp.addmessage(“Started batch geocoding job at: “ +
   time.ctime())

 #Create a dictionary with GP result status explanations
 statusDict = {0:”New”,1:”Submitted”,2:”Waiting”,
   3:”Executing”,4:” Succeeded”,5:”Failed”, \
   6:”Timed Out”, 7:”Canceling”, 8:”Canceled”,
   9:”Deleting”,10:”Deleted”}

 #We will append the results to an existing template
   feature class; truncate it first
 outFC = r”R:\Output.gdb\Customers”
 gp.addmessage(“Truncating output feature class: “ + outFC)
 truncResult = gp.deleterows_management(outFC)
 truncResult = gp.getcount_management(outFC)
 gp.addmessage(“Truncation result is: “
   +str(truncResult.getoutput(0)) + “ rows”)
```

```
#This script is in the Toolshare structure, go pick up the
  connection and input table
toolSharePath = os.path.dirname(sys.path[0])
gp.addmessage(“Toolshare path is: “ + toolSharePath)
oledbWorkspace = toolSharePath + r”\Connections\AGSUSER on
  BRUCEH.odc”
#The fully qualified table name will be source DBMS
  dependent.
#Edit this next line appropriately.
#Also ensure the table you reference agrees with the data
  source in the GP model
inputTable = oledbWorkspace + r”\dbo.FiveMillion”
gp.addmessage(“Reading table: “ +
  gp.parsetablename(inputTable).split(“,”)[2])

#Get the part count desired. This will be a multiple of
  available CPU cores.
#partCount = 8
partCount = int(gp.getparameter(0))
gp.addmessage(“Splitting input table into “ +
  str(partCount) + “ parts.”)

#Count the input table rows
result = gp.getcount_management(inputTable)
count = int(result.getoutput(0))
gp.addmessage(“Table “ + inputTable + “ has rowcount “ +
  str(count))

#Our GP service makes a query table on the 'ID' field, so
  the data must
#be appropriately structured
#Calculate the intervals suitable for a query ID >= <n>
  AND ID < <n>
intervals = range(1,count,int(count/partCount))
if len(intervals) > partCount:
intervals[-1] = count + 1
else:
intervals.append(count+1)

#Create the queries for each table part.
#The GP service will copy rows into the scratch workspace
  so delimit appropriately
dFName = gp.addfielddelimiters(gp.scratchworkspace,”ID”)
queryList = []
for i in range(len(intervals)-1):
queryList.append(dFName + “ >= “ + str(intervals[i]) + “
  AND “ \
+ dFName + “ < “ + str(intervals[i+1]))
gp.addmessage(“Built query: “ + queryList[i])
```

```
#Send each table part to the geocoding service
resultList, appendedList = [],[]
gp.addtoolbox("http://eslsrv23/arcgis/services;
  GeocodingBestPractice/ServerGeocode")
for q in queryList:
gp.addmessage("Sending query " + str(q) + " to the geocode
  service")
result = gp.GeocodeCustomers_ServerGeocode(str(q))
resultList.append(result)

#Wait until all the query geocoding results have begun
  turning up in job directories
numGeocoding = 0
resultPrefix = r"C:\arcgisserver\arcgisjobs\
  geocodingbestpractice\servergeocode_gpserver" \ + "\\"
resultSuffix = r"\scratch\scratch.gdb\Geocode_Result"
gp.addmessage("\n")
while (numGeocoding < partCount) and (time.clock() -
  startTime < 360):
gp.addmessage("Waiting for all geocoding processes to
  start..")
numGeocoding = 0
time.sleep(20)
for r in resultList:
resultPath = resultPrefix + r.resultid + resultSuffix
resultExists = gp.exists(resultPath)
gp.addmessage("Result " + str(r.resultID) + " exists?:" +
  str(resultExists))
if resultExists:
numGeocoding = numGeocoding + 1
gp.addmessage(str(numGeocoding) + " of " + str(partCount)
  + " parts now geocoding\n")

#As results turn up append them to the target feature
  class
endStatus = 0
while endStatus < 4:
loopStart = time.clock()
statusList = []
resultPath = ""
#For each part completing in this interval, get the result
  path
for r in resultList:
thisStatus = r.status
statusList.append(thisStatus)
gp.addmessage("Result "+ str(r.resultID) + \
" has status \'" + statusDict[thisStatus]+"\'")
if thisStatus == 4 and appendedList.count(r.resultid) ==
  0:
gp.addmessage("Result " + str(r.resultID) + " ready to
  append")
```

```
 resultPath = resultPath + resultPrefix + str(r.resultID) +
   resultSuffix + “;”
 appendedList.append(r.resultID)
 #Append anything that has completed. Remove the spatial
   index first.
 #Append will restore it.
 if len(resultPath) <> 0:
 resultPath = resultPath.rstrip(“;”)
 gp.addmessage(“\nAppending from “ + resultPath)
 outDesc = gp.describe(outFC)
 if outDesc.hasspatialindex:
 gp.removespatialindex_management(outFC)
 appendResult = gp.append_management
   (resultPath,outFC,”NO_TEST”)
 gp.addmessage(“Append complete\n”)
 else:
 loopEnd = time.clock()
 loopDuration = int(loopEnd - loopStart)
 if (loopDuration) < 60:
 gp.addmessage(str(time.ctime()) + “ Waiting balance of 60
   seconds: = “ \
 + str(60 - loopDuration) + “ sec.\n”)
 time.sleep(60 - loopDuration)
 statusList.sort()
 endStatus = statusList[0]

 gp.addmessage (“\nLast geocode/append process complete.”)
 appendResult = gp.getcount_management(outFC)
 gp.addmessage(“Output feature class has: “
   +str(appendResult.getoutput(0)) + “ rows”)

 endTime = time.clock()
 gp.addmessage(“Finished batch geocoding job at: “ +
   time.ctime())
 gp.addmessage(“Net performance was “ + \
 str(int(1.0 * count / (endTime - startTime) * 3600.0)) + “
   records per hour.”)
 gp = None
except:
 tb = sys.exc_info()[2]
 tbinfo = traceback.format_tb(tb)[0]
 pymsg = “PYTHON ERRORS:\nTraceback Info:\n” + tbinfo +
   “\nError Info:\n “ + \
 str(sys.exc_type)+ “: “ + str(sys.exc_value) + “\n”
 print pymsg
 gp.AddError(pymsg)

 msgs = “GP ERRORS:\n” + gp.GetMessages(2) + “\n”
 print msgs
 gp.AddError(msgs)
```

Example Message Stream from the Script Tool

```
Executing: RunBatchGeocode 8
Start Time: Thu Feb 26 08:25:46 2009
Running script RunBatchGeocode...
Started batch geocoding job at: Thu Feb 26 08:25:46 2009
Truncating output feature class: R:\Output.gdb\Customers
Truncation result is: 0 rows
Toolshare path is: C:\arcgisserver\Toolshare
Reading table: C:\arcgisserver\Toolshare\
  Connections\AGSUSER on BRUCEH.odc\dbo.FiveMillion
Splitting input table into 8 parts.
Table C:\arcgisserver\Toolshare\Connections\AGSUSER on
  BRUCEH.odc\dbo.FiveMillion has rowcount 5000000
Built query: “ID” >= 1 AND “ID” < 625001
Built query: “ID” >= 625001 AND “ID” < 1250001
Built query: “ID” >= 1250001 AND “ID” < 1875001
Built query: “ID” >= 1875001 AND “ID” < 2500001
Built query: “ID” >= 2500001 AND “ID” < 3125001
Built query: “ID” >= 3125001 AND “ID” < 3750001
Built query: “ID” >= 3750001 AND “ID” < 4375001
Built query: “ID” >= 4375001 AND “ID” < 5000001
Sending query “ID” >= 1 AND “ID” < 625001 to the geocode
  service
Sending query “ID” >= 625001 AND “ID” < 1250001 to the
  geocode service
Sending query “ID” >= 1250001 AND “ID” < 1875001 to the
  geocode service
Sending query “ID” >= 1875001 AND “ID” < 2500001 to the
  geocode service
Sending query “ID” >= 2500001 AND “ID” < 3125001 to the
  geocode service
Sending query “ID” >= 3125001 AND “ID” < 3750001 to the
  geocode service
Sending query “ID” >= 3750001 AND “ID” < 4375001 to the
  geocode service
Sending query “ID” >= 4375001 AND “ID” < 5000001 to the
  geocode service

Waiting for all geocoding processes to start…
Result j478b84f67a1242d3b60ae864bd802b4a exists?:False
Result je31b3ff90f634a1fa533965c16096bb2 exists?:False
Result jb6ad2721812e459981ba3b5bd6d3daaa exists?:False
Result j0d8e6eb2d66e47beaca123580221670b exists?:False
Result j30335c50ae3a4a549f053142b8b29182 exists?:False
Result jc632b67671a24e0ca2e9950ddd12d974 exists?:False
Result j5974490215994 11295cad0aaa5a6ede1 exists?:False
Result ja1f94a4a827748669af71d49d08c4865 exists?:False
0 of 8 parts now geocoding
```

```
Waiting for all geocoding processes to start…
Result j478b84f67a1242d3b60ae864bd802b4a exists?:False
Result je31b3ff90f634a1fa533965c16096bb2 exists?:False
Result jb6ad2721812e459981ba3b5bd6d3daaa exists?:False
Result j0d8e6eb2d66e47beaca123580221670b exists?:False
Result j30335c50ae3a4a549f053142b8b29182 exists?:False
Result jc632b67671a24e0ca2e9950ddd12d974 exists?:False
Result j597449021599411295cad0aaa5a6ede1 exists?:False
Result ja1f94a4a827748669af71d49d08c4865 exists?:False
0 of 8 parts now geocoding

Waiting for all geocoding processes to start…
Result j478b84f67a1242d3b60ae864bd802b4a exists?:False
Result je31b3ff90f634a1fa533965c16096bb2 exists?:False
Result jb6ad2721812e459981ba3b5bd6d3daaa exists?:False
Result j0d8e6eb2d66e47beaca123580221670b exists?:False
Result j30335c50ae3a4a549f053142b8b29182 exists?:False
Result jc632b67671a24e0ca2e9950ddd12d974 exists?:False
Result j597449021599411295cad0aaa5a6ede1 exists?:False
Result ja1f94a4a827748669af71d49d08c4865 exists?:False
0 of 8 parts now geocoding

Waiting for all geocoding processes to start…
Result j478b84f67a1242d3b60ae864bd802b4a exists?:True
Result je31b3ff90f634a1fa533965c16096bb2 exists?:False
Result jb6ad2721812e459981ba3b5bd6d3daaa exists?:False
Result j0d8e6eb2d66e47beaca123580221670b exists?:False
Result j30335c50ae3a4a549f053142b8b29182 exists?:False
Result jc632b67671a24e0ca2e9950ddd12d974 exists?:False
Result j597449021599411295cad0aaa5a6ede1 exists?:False
Result ja1f94a4a827748669af71d49d08c4865 exists?:False
1 of 8 parts now geocoding

Waiting for all geocoding processes to start…
Result j478b84f67a1242d3b60ae864bd802b4a exists?:True
Result je31b3ff90f634a1fa533965c16096bb2 exists?:False
Result jb6ad2721812e459981ba3b5bd6d3daaa exists?:False
Result j0d8e6eb2d66e47beaca123580221670b exists?:False
Result j30335c50ae3a4a549f053142b8b29182 exists?:False
Result jc632b67671a24e0ca2e9950ddd12d974 exists?:False
Result j597449021599411295cad0aaa5a6ede1 exists?:False
Result ja1f94a4a827748669af71d49d08c4865 exists?:False
1 of 8 parts now geocoding

Waiting for all geocoding processes to start…
Result j478b84f67a1242d3b60ae864bd802b4a exists?:True
Result je31b3ff90f634a1fa533965c16096bb2 exists?:False
Result jb6ad2721812e459981ba3b5bd6d3daaa exists?:False
Result j0d8e6eb2d66e47beaca123580221670b exists?:False
Result j30335c50ae3a4a549f053142b8b29182 exists?:False
Result jc632b67671a24e0ca2e9950ddd12d974 exists?:False
Result j597449021599411295cad0aaa5a6ede1 exists?:False
Result ja1f94a4a827748669af71d49d08c4865 exists?:False
1 of 8 parts now geocoding

Waiting for all geocoding processes to start…
Result j478b84f67a1242d3b60ae864bd802b4a exists?:True
Result je31b3ff90f634a1fa533965c16096bb2 exists?:True
Result jb6ad2721812e459981ba3b5bd6d3daaa exists?:True
Result j0d8e6eb2d66e47beaca123580221670b exists?:True
Result j30335c50ae3a4a549f053142b8b29182 exists?:True
Result jc632b67671a24e0ca2e9950ddd12d974 exists?:True
Result j597449021599411295cad0aaa5a6ede1 exists?:True
Result ja1f94a4a827748669af71d49d08c4865 exists?:True
8 of 8 parts now geocoding

Result j478b84f67a1242d3b60ae864bd802b4a has status
  'Executing'
Result je31b3ff90f634a1fa533965c16096bb2 has status
  'Executing'
Result jb6ad2721812e459981ba3b5bd6d3daaa has status
  'Executing'
Result j0d8e6eb2d66e47beaca123580221670b has status
  'Executing'
Result j30335c50ae3a4a549f053142b8b29182 has status
  'Executing'
Result jc632b67671a24e0ca2e9950ddd12d974 has status
  'Executing'
Result j597449021599411295cad0aaa5a6ede1 has status
  'Executing'
Result ja1f94a4a827748669af71d49d08c4865 has status
  'Executing'
Thu Feb 26 08:29:52 2009 Waiting balance of 60 seconds: =
  59 sec.

Result j478b84f67a1242d3b60ae864bd802b4a has status
  'Executing'
Result je31b3ff90f634a1fa533965c16096bb2 has status
  'Executing'
Result jb6ad2721812e459981ba3b5bd6d3daaa has status
  'Executing'
Result j0d8e6eb2d66e47beaca123580221670b has status
  'Executing'
Result j30335c50ae3a4a549f053142b8b29182 has status
  'Executing'
Result jc632b67671a24e0ca2e9950ddd12d974 has status
  'Executing'
Result j597449021599411295cad0aaa5a6ede1 has status
  'Executing'
Result ja1f94a4a827748669af71d49d08c4865 has status
 'Executing'
Thu Feb 26 08:30:52 2009 Waiting balance of 60 seconds: =
  60 sec.
```

```
Result j478b84f67a1242d3b60ae864bd802b4a has status
  'Executing'
Result je31b3ff90f634a1fa533965c16096bb2 has status
  'Executing'
Result jb6ad2721812e459981ba3b5bd6d3daaa has status
  'Executing'
Result j0d8e6eb2d66e47beaca123580221670b has status
  'Executing'
Result j30335c50ae3a4a549f053142b8b29182 has status
  'Executing'
Result jc632b67671a24e0ca2e9950ddd12d974 has status
  'Executing'
Result j5974490215994112 95cad0aaa5a6ede1 has status
  'Executing'
Result ja1f94a4a827748669af71d49d08c4865 has status
  'Executing'
Thu Feb 26 08:31:54 2009 Waiting balance of 60 seconds: =
  59 sec.

Result j478b84f67a1242d3b60ae864bd802b4a has status
  'Executing'
Result je31b3ff90f634a1fa533965c16096bb2 has status
  'Executing'
Result jb6ad2721812e459981ba3b5bd6d3daaa has status
  'Executing'
Result j0d8e6eb2d66e47beaca123580221670b has status
  'Executing'
Result j30335c50ae3a4a549f053142b8b29182 has status
  'Executing'
Result jc632b67671a24e0ca2e9950ddd12d974 has status
  'Executing'
Result j59744902159941129 5cad0aaa5a6ede1 has status
  'Executing'
Result ja1f94a4a827748669af71d49d08c4865 has status
  'Executing'
Thu Feb 26 08:32:54 2009 Waiting balance of 60 seconds: =
  59 sec.
```

```
Result j478b84f67a1242d3b60ae864bd802b4a has status
  'Executing'
Result je31b3ff90f634a1fa533965c16096bb2 has status
  'Executing'
Result jb6ad2721812e459981ba3b5bd6d3daaa has status
  'Executing'
Result j0d8e6eb2d66e47beaca123580221670b has status
  'Executing'
Result j30335c50ae3a4a549f053142b8b29182 has status
  'Executing'
Result jc632b67671a24e0ca2e9950ddd12d974 has status
  'Executing'
Result j597449021599411295cad0aaa5a6ede1 has status
  'Executing'
Result ja1f94a4a827748669af71d49d08c4865 has status
  'Executing'
Thu Feb 26 08:33:55 2009 Waiting balance of 60 seconds: =
  59 sec.

Result j478b84f67a1242d3b60ae864bd802b4a has status
  'Executing'
Result je31b3ff90f634a1fa533965c16096bb2 has status
  'Executing'
Result jb6ad2721812e459981ba3b5bd6d3daaa has status
  'Executing'
Result j0d8e6eb2d66e47beaca123580221670b has status
  'Executing'
Result j30335c50ae3a4a549f053142b8b29182 has status
  'Executing'
Result jc632b67671a24e0ca2e9950ddd12d974 has status
  'Executing'
Result j597449021599411295cad0aaa5a6ede1 has status
  'Executing'
Result ja1f94a4a827748669af71d49d08c4865 has status
  'Executing'
Thu Feb 26 08:34:56 2009 Waiting balance of 60 seconds: =
  59 sec.

Result j478b84f67a1242d3b60ae864bd802b4a has status
  'Executing'
Result je31b3ff90f634a1fa533965c16096bb2 has status
  'Executing'
Result jb6ad2721812e459981ba3b5bd6d3daaa has status
  'Executing'
Result j0d8e6eb2d66e47beaca123580221670b has status
  'Executing'
Result j30335c50ae3a4a549f053142b8b29182 has status
  'Executing'
Result jc632b67671a24e0ca2e9950ddd12d974 has status
  'Executing'
Result j597449021599411295cad0aaa5a6ede1 has status
  'Executing'
```

```
Result ja1f94a4a827748669af71d49d08c4865 has status
  'Executing'
Thu Feb 26 08:35:56 2009 Waiting balance of 60 seconds: =
  59 sec.

Result j478b84f67a1242d3b60ae864bd802b4a has status
  'Executing'
Result je31b3ff90f634a1fa533965c16096bb2 has status
  'Executing'
Result jb6ad2721812e459981ba3b5bd6d3daaa has status
  'Executing'
Result j0d8e6eb2d66e47beaca123580221670b has status
  'Succeeded'
Result j0d8e6eb2d66e47beaca123580221670b ready to append
Result j30335c50ae3a4a549f053142b8b29182 has status
  'Succeeded'
Result j30335c50ae3a4a549f053142b8b29182 ready to append
Result jc632b67671a24e0ca2e9950ddd12d974 has status
  'Executing'
Result j597449021599411295cad0aaa5a6ede1 has status
  'Executing'
Result ja1f94a4a827748669af71d49d08c4865 has status
  'Executing'

Appending from
C:\arcgisserver\arcgisjobs\geocodingbestpractice\servergeoc
  ode_gpserver\j0d8e6eb2d66e47beaca123580221670b\scratch\sc
  ratch.gdb\Geocode_Result;C:\arcgisserver\arcgisjobs\geoco
  dingbestpractice\servergeocode_gpserver\j30335c50ae3a4a54
  9f053142b8b29182\scratch\scratch.gdb\Geocode_Result
Append complete

Result j478b84f67a1242d3b60ae864bd802b4a has status
  'Succeeded'
Result j478b84f67a1242d3b60ae864bd802b4a ready to append
Result je31b3ff90f634a1fa533965c16096bb2 has status
  'Succeeded'
Result je31b3ff90f634a1fa533965c16096bb2 ready to append
Result jb6ad2721812e459981ba3b5bd6d3daaa has status
  'Succeeded'
Result jb6ad2721812e459981ba3b5bd6d3daaa ready to append
Result j0d8e6eb2d66e47beaca123580221670b has status
  'Succeeded'
Result j30335c50ae3a4a549f053142b8b29182 has status
  'Succeeded'
Result jc632b67671a24e0ca2e9950ddd12d974 has status
  'Succeeded'
Result jc632b67671a24e0ca2e9950ddd12d974 ready to append
Result j597449021599411295cad0aaa5a6ede1 has status
  'Executing'
Result ja1f94a4a827748669af71d49d08c4865 has status
  'Succeeded'
```

```
Result ja1f94a4a827748669af71d49d08c4865 ready to append

Appending from
C:\arcgisserver\arcgisjobs\geocodingbestpractice\servergeoc
   ode_gpserver\j478b84f67a1242d3b60ae864bd802b4a\scratch\sc
   ratch.gdb\Geocode_Result;C:\arcgisserver\arcgisjobs\geoco
   dingbestpractice\servergeocode_gpserver\je31b3ff90f634a1f
   a533965c16096bb2\scratch\scratch.gdb\Geocode_Result;C:\ar
   cgisserver\arcgisjobs\geocodingbestpractice\servergeocode
   _gpserver\jb6ad2721812e459981ba3b5bd6d3daaa\scratch\scrat
   ch.gdb\Geocode_Result;C:\arcgisserver\arcgisjobs\geocodin
   gbestpractice\servergeocode_gpserver\jc632b67671a24e0ca2e
   9950ddd12d974\scratch\scratch.gdb\Geocode_Result;C:\arcgi
   sserver\arcgisjobs\geocodingbestpractice\servergeocode_gp
   server\ja1f94a4a827748669af71d49d08c4865\scratch\scratch.
   gdb\Geocode_Result
Append complete

Result j478b84f67a1242d3b60ae864bd802b4a has status
  'Succeeded'
Result je31b3ff90f634a1fa533965c16096bb2 has status
  'Succeeded'
Result jb6ad2721812e459981ba3b5bd6d3daaa has status
  'Succeeded'
Result j0d8e6eb2d66e47beaca123580221670b has status
  'Succeeded'
Result j30335c50ae3a4a549f053142b8b29182 has status
  'Succeeded'
Result jc632b67671a24e0ca2e9950ddd12d974 has status
  'Succeeded'
Result j597449021599411295cad0aaa5a6ede1 has status
  'Succeeded'
Result j597449021599411295cad0aaa5a6ede1 ready to append
Result ja1f94a4a827748669af71d49d08c4865 has status
  'Succeeded'

Appending from
C:\arcgisserver\arcgisjobs\geocodingbestpractice\servergeoc
   ode_gpserver\j597449021599411295cad0aaa5a6ede1\scratch\sc
   ratch.gdb\Geocode_Result
Append complete

Last geocode/append process complete.
Output feature class has: 5000000 rows
Finished batch geocoding job at: Thu Feb 26 08:45:45 2009
Net performance was 15016112 records per hour.
Completed script RunBatchGeocode...
Executed (RunBatchGeocode) successfully.
End Time: Thu Feb 26 08:45:45 2009 (Elapsed Time: 19
  minutes 59 seconds)
```

About ESRI

For four decades, ESRI has been helping people make better decisions through management and analysis of geographic information. Our culturally diverse staff work with our business partners and hundreds of thousands of people who use GIS to make a difference in our world.

A full-service GIS company, ESRI offers support for implementing GIS technology from the desktop to enterprise-wide servers, online services, and mobile devices. GIS solutions are flexible and customizable to meet the needs of all our users.

Our Focus

At ESRI, we focus on promoting the value of GIS and its applications throughout the world and pay close attention to our users' needs. Our software development and services respond to our customers with products that are easy to use, flexible, and integrated. Our technology is multidisciplinary, productive, and valuable to our users.

We have a strong commitment to educating our customers through ESRI's various training programs. ESRI is a socially conscious business and invests heavily in issues regarding education, conservation, sustainable development, and humanitarian affairs.

Contact ESRI

1-800-GIS-XPRT (1-800-447-9778)
Phone: 909-793-2853
Fax: 909-793-5953
info@esri.com
www.esri.com

Offices worldwide
www.esri.com/locations

ESRI
380 New York Street
Redlands, California
92373-8100 USA

GSA Contract Holder
Contract GS-35F-5086H

Printed in USA