Synchronizing Smallworld Data with ArcGIS

By Mark Cederholm

Abstract:

In order to make the rich suite of mapping tools present in ArcGIS available to a Smallworld shop, it is necessary to expose the data. After a review of technological options, it was decided that maximum flexibility and performance could be achieved by mirroring the data model in a geodatabase. Synchronization is accomplished through periodic incremental updates. This paper will describe a pilot project to publish data to field users using ArcGIS Publisher. Future directions include bidirectional synchronization in order to utilize landbase maintenance tools in ArcGIS.

1. USING ARCGIS IN A SMALLWORLD SHOP

UniSource Energy Service's Gas division has used Smallworld as its enterprise GIS since the late 1990s. At the same time, however, ESRI offered products that made mapmaking easier and more intuitive and produced better output. The conundrum facing us was: how can we maintain our data in Smallworld and at the same time take advantage of ESRI's mapmaking tools?

Originally, our strategy was to export shapefiles from Smallworld for use in ArcView. Shapefiles are convenient in that they are portable and widely supported, but they need to be refreshed as the data in Smallworld is updated. Before long, we had who knows how many versions of shapefiles floating around in users' directories. Also, shapefiles have severe limitations in modeling our data: they support a limited number of geometric types (no annotations or dimensions), they do not allow more than one geometry field per record, and join fields are difficult to mimic.

Since then, ESRI has improved and matured its geodatabase technology, especially with the introduction of topology in version 8.3 and annotation classes in version 9. Because of that, we decided to try to find an alternative to shapefiles to represent our data in ArcGIS. Table 1 summarizes the advantages and disadvantages of the options considered.

After evaluating ArcSDE, I considered it the best technical solution because of its superior performance and support for multiple writers. I determined that as of version 9 an ArcSDE geodatabase could adequately represent our data model, albeit with some workarounds. For example, objects with multiple geometry fields can be represented by multiple objects joined via relationship classes. And annotations and dimensions are thankfully supported.

Data synchronization would be a problem, but not an insoluble one. And although an ArcSDE geodatabase is not accessible to non-networked users, data may easily be checked out into a personal geodatabase and refreshed periodically. Unfortunately, this option was ultimately excluded because we couldn't get approval for the licensing.

Table 1. Options for representing Smallworld data in ArcGIS

Option	**Advantages**	**Problems**
Shapefile	• Portability	• Data synchronization • Inadequate data model support
ArcSDE	• Best performance • Data model support • Multiple writers	• Data synchronization • Not portable • Expensive
Custom OLE DB Provider	• Live data access	• Inadequate data model support • Not portable • Read only • Performance?
Personal Geodatabase	• Portability • Data model support	• Data synchronization • Multiple writers

As a different option, I investigated the possibility of creating a custom OLE DB provider for Smallworld. This would have the obvious advantage of allowing access to live data. It didn't take me long, however, to discern a number of technical disadvantages. The biggest disadvantage is the limitation of the OGIS simple feature standard. How do you model multiple geometries? How do you navigate join fields? How do you represent annotations and dimensions?

Other considerations included how to get data to field users without a network connection. Also, the approach is read-only: if we wanted to perform edits in ArcGIS, how could we get them back to Smallworld? [I did have some concerns about performance, but during the course of developing my Smallworld/ArcGIS interface I was able to benchmark an average of about 800 points/second, 500 lines/second, and 300 polygons/second, good enough for tolerable rendering.]

Ultimately I decided to pursue the personal geodatabase option. Personal geodatabases can model anything an ArcSDE geodatabase can; indeed, one can be converted right into the other. Personal geodatabases have a 2 GB size limitation, but that can be overcome by splitting up data into multiple sets by category or region. In addition, personal geodatabases can be copied to laptops for field users.

The two major problems surrounding the use of personal geodatabases are data synchronization with Smallworld and supporting multiple writers. Though there are some practical considerations and limitations, I have developed a solution to both problems that seems to be working well for our (admittedly fairly small) GIS shop.

2. SOLVING THE SYNCHRONIZATION PROBLEM

The initial COM-based Smallworld/ArcGIS interface that I developed (SWReader) reads records from an active Smallworld session and packages them into ArcObjects. Thus, one could write a standalone ArcObjects application, or a VBA script in ArcCatalog, that could loop through the objects in a table, browse all geometry and attribute fields, navigate join fields, and reconstruct the data and relationships on the ArcGIS side.

The disadvantage of this interface is that it was designed to export entire datasets, not perform incremental updates. Also, it soon became apparent that sometime down the line we would want to be able to create data in ArcGIS and send it back to Smallworld. The next interface that I designed (SWArcExchange) allows bidirectional incremental updates. Both interfaces are available at ESRI's ArcScripts site.

In order for the data to be properly synchronized, there must be a consistent means to identify features uniquely and to determine when they were last created or updated. In addition, a mechanism for tracking deletions must be in place. Because Smallworld can key records in any number of ways, I added a single ID field to each object that would be populated by a string representation of its key. I also added a timestamp field to each object containing the date and time of its creation or last update:

```
Name                Type
--------------------------------
synch_id            ds_charci 64
synch_timestamp     ds_time
```

For each Smallworld partition, I also created a deletion tracking table:

```
Table name:  deletion_table
Name                Type
---------------------------------------
id                  ds_uint (key field)
synch_timestamp     ds_time
synch_id            ds_charci 64
parent_table        ds_charci 30
```

Each object with synch attributes has an insert, update, and delete trigger. The insert trigger populates the synch id and timestamp, the update trigger updates the timestamp, and the delete trigger adds a record to the deletion table.

On the ArcGIS side, I created personal geodatabases (one for each Smallworld partition) that matched our Smallworld data model as closely as possible. As previously mentioned, some changes had to be made for objects with multiple geometry fields. For example, parcels and address points which have annotation fields simply had the annotation replaced with auto labeling. Service points, which have carefully placed annotation, had a related annotation feature added. GPS data objects, which have both point and line fields, were split into separate features.

The personal geodatabases, which I will refer to as "central geodatabases", are read-only to ArcGIS users, and placed on a network drive so that various mapping and reporting applications can all access them. This leads to some performance issues, but also helps prevent the proliferation of various versions of the data.

After initially loading Smallworld data into the central geodatabases, further updates are transmitted through a periodic incremental update process. A scheduled task runs a batch file which launches a Smallworld synchronization application. The application starts the SWArcExchange server and then launches an ArcObjects client application which performs the synchronization process. The process queries for features that were deleted, created, or updated since the last synchronization, and then it performs comparable operations on the geodatabase. The synch cycle is carefully orchestrated so that child objects are regressively deleted before parent objects, and parent objects are created before the child objects which are then joined to them.

Some practical problems and considerations arose from synchronization scheduling, which will be discussed elsewhere.

The next problem involved how to get data from ArcGIS back to Smallworld. The central geodatabases are read-only to prevent accidental tinkering, and in any case will not support multiple writers. The workaround to this is to have the users create blank copies of the geodatabases. These "edit geodatabases" can then be populated with features.

Initially, I worked along the assumption that features created in an edit database would simply be "posted" to create new records Smallworld: Smallworld itself would assign the synch id and timestamp. I soon realized that a user may want to be able to edit or delete features in ArcMap (for example, as part of an ongoing design) and see those changes reflected in Smallworld. The problem I immediately saw with this is: how do you assure that separate users with separate edit geodatabases will create truly unique synch IDs in ArcMap?

Finally, I hit upon the idea of having the edit event handler assign a GUID to the current ArcMap session when it opens. A GUID, or Globally Unique Identifier, is a value generated by a Windows API call which can be relied upon with a great deal of confidence to be unique for anyone, anywhere, at any time. The synch ID for each new feature created then becomes the session GUID plus the feature object ID. The edit event handler handles creation, update, and deletion events just as the corresponding triggers do in Smallworld.

The synchronization process, then, is as follows. The user launches a Smallworld session, navigates to a writable alternative, and starts the SWArcExchange server. In the ArcMap session, a button fires a script which loops through all the feature classes in the edit geodatabase. For each feature class, the script finds entries in the deletion table, deleting the corresponding features in Smallworld, and then loops through all the existing features in the feature class. If the comparable feature doesn't exist in Smallworld, it's created and assigned the ArcGIS synch id and timestamp. If it exists and the timestamp is the same, nothing happens. If the timestamp in Smallworld is earlier than the one in ArcGIS, the feature in Smallworld is updated and assigned the ArcGIS timestamp. And if the timestamp in Smallworld is later, a conflict is flagged.

Once the ArcGIS-Smallworld synchronization takes place, the user merges and posts the changes in Smallworld. Those changes then make their way to the central geodatabases at the next Smallworld-ArcGIS synchronization, and the process comes full circle. One important aspect of this approach is that you cannot update or delete features that were not created in your edit geodatabase. However, it should be possible to import select features from a central geodatabase into an edit geodatabase for editing: a process will need to be developed to facilitate and regulate this.

3. THE PILOT IMPLEMENTATION

For our pilot project, we considered getting data to gas facility locators in the field. In the past, we put shapefiles and an ArcExplorer project onto a CD-ROM for distribution, but the shapefiles were not being updated often enough (the consensus that arose was that the data should be refreshed at least once every two weeks). Another problem was the inability to display annotation: much of our annotation (especially that of customers and streets) is carefully placed for clarity.

When I was able to demonstrate that an ArcMap project running off a geodatabase could closely resemble a Smallworld session in appearance, with nearly identical display scales and symbology, we decided that the ArcGIS Publisher extension, coupled with ArcReader, would suit the needs of the users. Once the Smallworld-ArcGIS synchronization process was in place, we created and configured an ArcMap project with a look and feel that made the locators happy and published it to an ArcReader project. Then, I created a custom application using the ArcReader control and added some query forms allowing easier location of addresses, lot numbers, and street intersections. Finally, the custom ArcReader app was installed on the locators' laptops and a data update shortcut created, allowing locators to update the files on their laptops with a simple click.

4. PRACTICAL PROBLEMS AND CONSIDERATIONS

There are two major factors on the Smallworld side that can seriously impact the Smallworld-ArcGIS synchronization: 1) the number of edits to synchronize, and 2) how long edits remain in a user's alternative before they are posted.

Even though all feature classes in Smallworld and the central geodatabases have an index on the synch ID field, the process of finding and comparing records is time intensive. The more often the synchronization takes place, the fewer the edits needed to synchronize, and the shorter the process time. A very large number of edits at any time is problematic: there is a point at which the time it takes to synchronize a large number of records exceeds the time it takes to delete the entire feature set and reload it.

One must also prevent a situation where edits are posted in Smallworld that have a timestamp earlier than the search pattern used. Currently, there is a parameter in the synchronization code that can be used to specify the number of days to search prior to the last synchronization. However, that can greatly increase the number of records to find and compare and therefore increase performance time. In our shop, Smallworld users typically merge and post their work daily, if not more often, so this is rarely an issue for us.

In our situation, for the process to work most effectively, we follow these guidelines:

- Schedule the Smallworld-ArcGIS synchronization every night, in the largest time window allowable before it conflicts with other scheduled tasks.
- Merge and post all Smallworld edits before the end of the day.
- Do not search for timestamps earlier than the last synchronization.
- Where very large numbers of features are affected (such as updating a city or county landbase), remove deletion entries and reload the entire table into the central geodatabase.

One other problem that occasionally occurs is incomplete synchronization. The record counts of a couple of feature classes sometimes drift apart, and I'm still in the process of figuring out why that is happening. In the meantime, an occasional reload is a temporary solution.

An issue also exists for the ArcGIS-Smallworld synchronization in that very large numbers of edits such as replacing landbase should be posted directly without searching for corresponding features. A mechanism is in place to allow this. Another problem involves objects in Smallworld that, when updated, trigger updates on child objects, thereby creating timestamp conflicts. This problem is circumvented by forcing conflict resolution on child objects during the synchronization process. Doing this raises the danger of overriding "genuine" conflicts, but until we can find a better approach the risk is acceptable.

Finally, I discovered that certain processes in ArcGIS will create features without triggering a feature creation event (e.g. creating feature-linked annotation for selected features). Until a solution is found, the synchronization application rejects features lacking synch ids and alerts the user.

5. FUTURE DIRECTIONS

Our current landbase maintenance process has two components. AutoCAD drawings from contractors are rubbersheeted and cleaned up using TCI and AutoCAD, processed through FME into shapefiles, and then imported into Smallworld. Shapefiles from city and county agencies are reprojected and directly imported. ArcMap currently has the tools necessary to create landbase features from a variety of data sources. Coupling these with the ArcGIS-Smallworld synchronization tool should greatly improve the process and reduce the number of 3rd-party applications needed.

I am also in the process of building a project for creating and updating features from GPS data. Currently, GPS data goes to PathFinder Office, then to AutoCAD, then through FME to shapefiles, and finally into Smallworld. Hopefully, combining the GPS Analyst extension, some ArcMap feature creation tools, and the ArcGIS-Smallworld synchronization tool will allow us to simplify the steps and again reduce the number of 3rd-party applications needed.

Eventually, I plan to model our network topology and CP circuits in the geodatabase. Indeed, at this point there is no reason why all end user tasks cannot be done in ArcGIS, and that is our eventual goal. As this takes place, demand will increase for improving the performance and immediacy of applications using the central geodatabases, thereby strengthening our justification for acquiring ArcSDE. Then, we will need to re-evaluate the synchronization problem; hopefully, by that time all edit tasks will take place in ArcGIS, thereby simplifying the logistics.

6. CONCLUSION

In conclusion, the implementation and synchronization of personal geodatabases can be an effective approach to expose Smallworld data to ArcGIS for mapping and analysis. Furthermore, reverse synchronization allows the use of ArcGIS for editing. Close consideration must be paid, however, to the overall volume of edits, the timeliness of posting edits, and the consequent impact on synchronization tasks. Nonetheless, should all editing tasks migrate to ArcGIS, the problem could materially lessen.

NOTE: Example source code for the projects mentioned is available at http://www.pierssen.com/arcgis/misc.htm

Mark Cederholm, GISP
Operations Technology Integrator
UniSource Energy Services
2901 W Shamrell Blvd, # 110
Flagstaff, AZ 86001
Phone: 928-226-2235 Fax: 928-779-5338
E-mail: mcederholm@uesaz.com