Generalization of Multiple Scale Maps from a Single Master Database

Miguel Garriga, Geoff Baldwin

Abstract: This paper describes some of the work being done on map generalization at the American Automobile Association (AAA). AAA produces various map products at different scales for our members, including detailed city maps, vicinity maps, regional maps, and atlases. Our goal is to maintain all of the required geographic data in a single master database and to generate all of our map products from this one database. A closely related goal is make updates only once, directly to the master database, and to have those changes automatically appear in all derived maps the next time they are produced.

Introduction

This paper describes map generalization as it is being practiced at the American Automobile Association (AAA) for production of maps at various scales. We use both UNIX Arcinfo and the newer ArcGIS 8.1 desktop software for the NT. The main focus of this paper is Automated Extraction, the technique we use to pre-select which features should appear on a particular map product.

This paper is an update to a presentation given at the 1998 ESRI User Conference, "*Producing Multiple Scale Maps from a Single Master Database*" , available online at: http://www.esri.com/library/userconf/proc98/PROCEED/TO350/PAP327/P327.HTM. That paper described AAA's approach to map generalization at the start of our GIS map development effort. This update describes some of the changes in our generalization philosophy, and what our current thinking is for the future of map generalization at AAA.

Objectives for GIS Map Production - Mission Statements

GIS was brought to AAA with the following expectations and objectives:

- Create a seamless, nationwide master database to support all map production needs
- Produce multiple map products from a single master database
- Update features only in master database, synchronize changes to all map products
- Cartographic edits only on map products, not the master database
- Yearly updates to map products will be able to re-use last year's work, and also include all changes to the master database that might affect a given map product.
- Support electronic travel-related products, such as the Internet Triptik.

Other organizations using GIS for Cartography may share some or all of these goals. Individually, these goals seem logical and workable. No one objective seems that difficult, but taken together they form quite a challenge.

Ideal - Live Generalization of product from master database

An ideal generalization solution for AAA would be one where the master database contains detailed data and all map products are "live" views into that data. All generalization would occur on the fly, with no duplication of data. The system would not create new generalized datasets, but would make on the fly display calculations to display features from the master database according to specific rules for the desired map product. The map would be a specialized way of looking at the master data, without actually containing data.

To summarize, an ideal generalization environment for us would be as follows:

- The source data would be stored in a seamless nationwide (worldwide?) database such as SDE
- A map could be generalized on the fly, in real time, in its own projection with minimal interactive work required.
- There would be no duplication of data. The map would consist of features from the master database, generalized in real time to enhance their appearance or behavior on the map.
- The user would be able to make adjustments to the output of generalization to add, remove, or otherwise change the appearance of objects as needed.

So far we have not been able to achieve this ideal of live generalization. We have also not been able to create map products without duplicating data. The duplication of data is an issue for us mainly because duplicating features complicates the update of the master database and derived products. The scenario we want to get away from is where a change needs to be made to every individual map product a feature appears in. We would rather make the change only once, in the master layer, and have that change appear on all derived map products, including maps on a regular maintenance cycle.

We find that it is extremely difficult to create maps in different scales and projections directly out of the master database without duplicating data. At least we have not found a way to do it that works for a wide range of scales. Someday processing speeds and new functionality will allow us to achieve this goal of not duplicating data. In the meantime, we will have to develop a synchronization program to propagate changes that are made at the master level to all of the relevant product layers.

Evolution of our Approach to Generalization

Our map production procedures are a set of compromises that we have reached between our desire to have a single master database, our technical abilities, and performance of the software and hardware we use to do our work. Since we cannot generalize the master database on the fly, we compromise and create map-specific datasets as the output of generalization. This potentially complicates update scenarios, but this compromise is necessary in order to meet our map production schedules.

From ArcStorm Visibility files to SDE Product Layers

The visibility file concept is the closest we have come to not duplicating data for map products. When we started making maps with the GIS, we avoided duplicating data by using "visibility files" - INFO files containing unique feature IDs, which we used to create Arcplot selection sets (WRITESELECT and READSELECT) to control which features appeared on a map product. A single product layer per map was created to hold all features that needed to be displaced or altered in some way from their default properties in the master database. Each individual map view was projected on the fly using MAPPROJECTION, from the master projection in Albers to the specific product's projection and parameters.

The performance of selection sets and Mapprojection under ArcStorm was tolerable with UNIX Arcinfo 7.2.1. When we switched to Informix SDE and UNIX Arcinfo 8.0, Mapprojection's performance was noticeably slower, and deemed unacceptable by our cartographers and developers alike. We decided to shift our approach to generalization.

In order to improve the performance of the system with SDE, we did away with the old Visibility files and Mapprojection. Instead, we took the visibility file concept a step further. The visibility files used to be plain INFO files with one column - the unique ID of each feature in a map product. That was all we needed to keep track of which features in the master database should appear on a particular product. We decided to add a shape column, essentially turning the visibility file into a coverage or shapefile. Since we now had an actual coverage we could manipulate, we applied the product projection directly to the extracted data, doing away with the Mapprojection command, and this improved performance tremendously.

What we ended up with, instead of visibility files, was product layers in SDE for each map product. Cartographic displacements and other generalization processing could be done directly on the product layers. We decided to keep all attribution on the master database only, in order to prevent users from accidentally making master-level changes to the product layers.

Right now, product layers are just the shape, a unique id, and a symbol item; not much different than a CAD file. We are giving some thought to leaving the full attribution on them. This would give us greater flexibility and control over the appearance of features in a map product. Another advantage subsequent map revisions would be able to see exactly what was on the old map, including underlying attributes. We are still debating the merits of this change. The concern is that it will be difficult to coordinate edits to the master and product layers. We would have to develop some sort of synchronization program to propagate changes from the master database down to the product layers.

Ramping up to Generalization

Our approach to generalization has been gradual. Our source database is highly detailed, street level data from Navigation Technologies (NavTech) and Geographic Data Technologies (GDT), along with value-added AAA features and attribution. Generalization issues increase in complexity the further you get from the source data's scale or resolution. With some exceptions, we have approached generalization by starting with large scale maps that are closer to the source scale of the data, and over time have progressed to smaller scale maps.

Our first prototypes using GIS software (UNIX Arcinfo) were the Alabama/Georgia State maps, and the Atlanta CitiMap. Based on these early prototypes, we decided to focus our development efforts on creating a system for CitiMap production. The principal advantage of doing this was that CitiMaps were much closer to our source data scale, and therefore had fewer generalization issues. We used nearly all of the features in our source database for the city scale maps. Our generalization needs consisted of:

- Centerlines - Creating a centerline network out of divided roads, such as Interstates, which are digitized in both directions of travel.
- Weeding out features of interest (FOIs). Initially this was done manually, but we now have Automated Extraction routines to pre-select layers for each map type.
- Maplex for Text Placement.
- Displacing features as needed to ensure that they do not obscure each other. This is similar to the problem of text placement - accommodating lots of information on a map, preserving spatial relationships, yet moving things around a bit to better accommodate all the information.

Centerline Layer

Our data sources provide us with very detailed, high quality data. Interstates and other divided roads are digitized as separate arcs in each direction of travel. At most map scales, we represent divided highways with a single line symbol. We only show both directions of travel with separate linework on our downtown insets. Since the vendor provides doubly digitized arcs, not centerlines, we had to create the centerlines.

Although Arcinfo now has a CENTERLINE command which will create centerlines out of separately digitized directions of travel, this command was not available when we created our centerline layer. We contracted this work out, and received a centerline layer that required some interactive cleanup work. Most of the cleanup was at intersections, where centerlines sometimes went astray. Also, we did considerable work to create ramp extensions, so that existing rampwork would hook up to the new centerlines.

I want to emphasize that we did not use Arcinfo's CENTERLINE command, so these comments about cleanup may not be applicable to the CENTERLINE command.

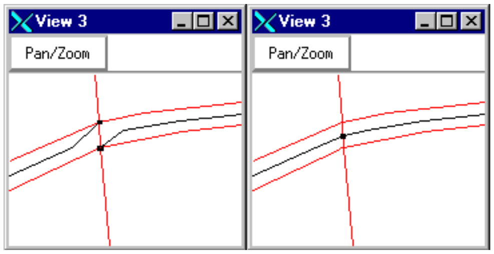

Figure 1. Fixing collapsed roads at intersections

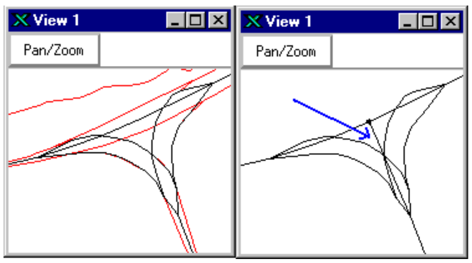

Figure 2. "T"-ing off intersections where doubly digitized roads meet

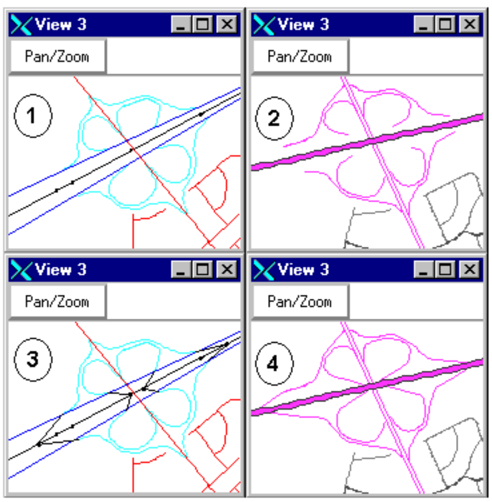

Figure 3. Extending Rampwork (1,2) to meet up with centerlines (3, 4)

Automated Extraction (Pre-selection)

Feature Matrix

At the start of the GIS project, a lot of effort was put into examining our existing map products and tracking what types of features went into each of the different products that we make. The result of this effort was the Feature Matrix, which lists individual feature types (roads, national parks, etc,) as rows and individual map inset types as the columns. An 'X' in a cell indicated that a feature did appear on that map product. In the early part of GIS map production, before automated Extraction, we relied on the Feature Matrix, essentially using the matrix to determine what features to manually extract from the database for each new GIS map product.

Turning the Feature Matrix into Extraction Rules

The Feature Matrix was the starting point for Automated Extraction. Although the matrix told us if a feature appeared on a product, it did not tell us all the parameters we would need to determine if a feature should appear on an inset. For example, a state map product includes lakes, but only lakes larger than a certain area, and that cutoff size is probably related to the scale of the map product, and what type of information that product is supposed to convey. So the next step for us was to develop extraction rules, elaborating on the basic information in the matrix, and specifying any parameters needed for inclusion in a product.

One of our senior cartographers was a key member in developing the Feature Matrix, and took on the task of writing the precise Extraction Rules. Because he helped develop the matrix, he had a good understanding of the logic behind why features occur on each map type.

We wrote the extraction rules in pseudo code, in such a way that a programmer could easily write code from it in whichever platform we ended up using (UNIX or NT). As a result of this effort, we created one set of "Extraction Rules" for each map series. Each set of documents listed all the extraction logic for up to 7 possible inset types per map, for each of 11 possible GIS layers.

Turning the Extraction Rules into Program Code

We coded Extraction Rules in AML because the performance for this type of SDE attribute selection logic was faster with UNIX Arcinfo than with ArcMap desktop, at least during the Beta test period. Now that Arcinfo 8.1 is final, we will revisit our approach and re-evaluate selection performance on the desktop.

During this coding phase, a programmer took the cartographer's Extraction Rules and translated them into AML code. We added an option to the map production user interface to allow cartographers to run the Automated Extraction routines on their own, as one of the first steps in their map production process. After Automated Extraction, the layers needed for the map are ready for the cartographer to use.

Evaluation of Automated Extraction

The automated extraction itself works fairly well, but maintaining changes to the rules and the code has been a challenge. As cartographers run the extractions, they find ways to improve the extraction. These suggestions are passed on to our senior cartographer, who modifies the rules, and then notifies the programmer, who then modifies the code. This may seem easy and straightforward, but it is a logistical challenge to keep up with all the changes and improvements to the rules in a production environment. Well-structured code and good communication between the programmer and the cartographer are essential for the success of this effort.

Selection from SDE

Ordinarily, we would prefer to do our spatial selection first, followed by our attribute selection. For instance, a map of Miami Beach has a fairly small spatial extent, relative to a nationwide database. It would be much faster to do a spatial select first, followed by an attribute query to get the features we want within that extent. The alternative, to do an attribute selection on the whole country and then a spatial select for the area we want, is extremely inefficient.

Unfortunately, in the version of Arcinfo we are using (7.2.1), you must do the attribute selection first (LAYERQUERY), and follow that with a spatial selection (LAYERSEARCH or LAYERFILTER). Starting in 8.0, there is a LAYERSEARCH ORDER option which will allow you to change this default and do spatial selects first, but unfortunately we are still using 7.2.1 because we found that SDE draws much faster with Arcinfo 7.2.1 than with Arcinfo 8.0.

Now that Arcinfo 8.1 is final, we will have to evaluate performance again (comparing 8.1 and 7.2.1). If Arcinfo 8.1 draws SDE as fast as Arcinfo 7.2.1, then we would be able to get back on track with the latest version of Arcinfo and take advantage of the LAYERSEARCH ORDER command to optimize our spatial and attribute selections.

Content Management

After automated extraction is run, our map coordinators check the results. If there are features that are needed, they can be retrieved from the master database, or from one of the working coverages used in Automated Extraction. While we would like to think that our automated extraction rules can create a perfect map every time, we know it cannot be perfect, and that some features will have to be manually added, and others removed after automated extraction. The bulk of this tuning is done right after extraction. Once the map coordinator has verified the content, map production can begin. We try to fix most of the content issues before starting map production, but inevitably issues arise, so we do some fine-tuning of content throughout the life of the map product.

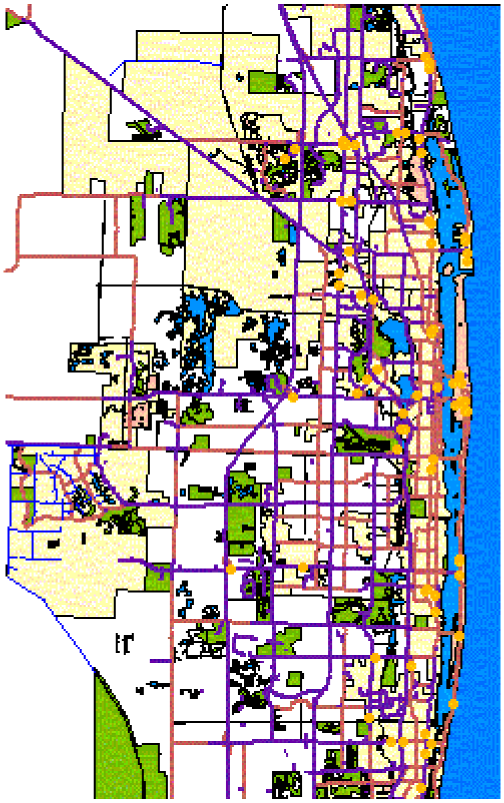

Figure 4. Results of Automated Extraction for Tourbook Spotting map for West Palm Beach. The default extraction is then manipulated by the cartographer as needed to produce the final map. Some features need to be added from master, others removed from the product.

Figure 5. Finished Tourbook Spotting map for West Palm Beach. The default extraction has been modified as needed, the map was finished with ArcMap.

Generalization requirements for different map types

This section outlines some of the different generalization needs for our products, specifically Sheet Maps, Tourbook Maps, Internet Triptik.

As mentioned earlier, the greater the difference between the source scale and the product scale, the more generalization work there is to be done. Our experience with generalization is a direct result of the types of maps we have developed, and the chronological order in which we have produced them. If we started our GIS map production system with smaller scale maps like the Regional maps rather than the large scale maps like the city maps, we might be saying different things today about generalization.

Sheet Map production

Sheet maps are large maps, typically with multiple views and indexes, that are folded to fit in a car's glove compartment. They include the CitiMap, Vicinity, State, and Regional Planning map series. AAA's sheet map production system runs on UNIX Arcinfo 7.2.1 and is based on ArcTools, with custom functionality added by AAA to create the custom look of our products.

We have recently started producing smaller scale Vicinity, State, and Regional maps, and are now encountering many more "opportunities" for generalization. Currently some generalization is being done manually, but we intend to automate as much as we can once we have identified the operations needed. We are still compiling information on the generalization requirements for these map products.

As the scale of the product becomes smaller, it is much more likely that multiple features will occupy the same space, and will need to be reconciled to be clearly visible. Some features would be lost completely without generalization. The only way to convey all the information is to abstract and generalize features, compromising real world locations for legibility.

Our Regional maps present new generalization challenges. The Southeastern States map is easily one third of the country. Most of the visitors to the southeast, particularly those who drive there, are from the northeast, and some from places as far away as Chicago and Canada. In order to accommodate these travelers, the back panel of our Southeastern States Regional Planning map extends as far north as Chicago, Detroit, Windsor in Ontario, and New York.

We are still investigating the generalization issues for producing Regional maps of this scale from a detailed, street-level database. The main challenge will be the enormous volume of data to filter.

Tourbook Map production with ArcMap

The Tourbook map production system is currently being developed with ArcGIS 8.1 on the NT desktop.

We needed a system that could make a large number of small maps easily and quickly. Rather than add more custom functionality to the UNIX Arcinfo system, which is no longer evolving as a product, we decided to see if the new ArcGIS 8.1 desktop tools could be used to create Tourbook maps. The desktop tools have a lot of built-in functionality that appear to surpass the custom tools we developed for UNIX.

Because the Tourbook maps cover a very wide range of scales, this presents an opportunity for us to make a lot of progress on our automated generalization procedures. The Tourbook map production system is in its early stages of development, so many issues related to generalization on the desktop are still being explored.

We are going ahead with cartographic production using ArcMap, but we are going continue to run Automated Extraction for Tourbook maps on UNIX because we believe the processing speed for the extraction is better on UNIX than with ArcMap. At least this was our conclusion during the ArcGIS 8.1 Beta period. Now that Arcinfo 8.1 is final, we will have to reassess, and see if our Automated Extraction can be done efficiently on the desktop.

We are also Beta testing the Adobe Illustrator Export from ArcMap, and the possibility of using Adobe Illustrator for finishing the Tourbook maps. Under this scenario, we would get the map as far as possible with ArcMap, and just use Adobe Illustrator to add graphic effects to the map.

Internet Triptik

The Internet Triptik (ITT) is primarily an Internet routing service rather than a map production system. ITT's needs and approach to map generalization is different than for AAA's paper map products. The primary purpose of the Internet Triptik is to route someone from an origin to a destination. In addition to showing the route itself, the Triptik maps show major roads in the vicinity of the route in order to properly orient the driver, and to allow some flexibility should the driver need to stray from the route.

The Internet Triptik relies on pre-generalized datasets for specific scale ranges. Depending on your scale, you will draw from one of a number of predefined datasets for specific scale ranges. This approach is quite different from our paper map products, and is driven by the need for fast performance over the Internet.

Generalization Issues for Various Paper Map Types

Some of the generalization issues we have identified for various map types include:

- Line simplification. We do simplify lines for the Internet Triptik layers, but we don't always simplify lines for paper products. In general, we simplify only where necessary to improve performance of displays, not necessarily for smoother appearance.

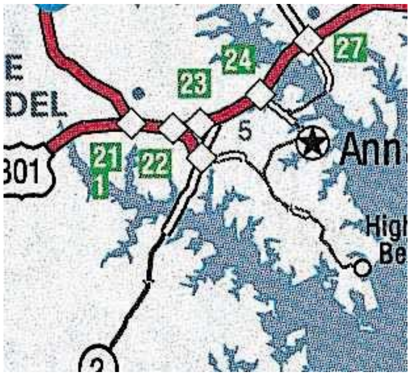

Figure 6. Line detail of Chesapeake Bay at vicinity scale, from the Washington DC Beltway and Vicinity map.

- Road Convergence. As they converge on urban areas, major roads tend to occlude each other. Due to the scale of the map and the thickness of the road line symbol at that scale, a single line could obscure a number of nearby roadways that run parallel or tangent. For example, Interstate 95 and the Florida Turnpike converge and run within a few hundred feet of each other for several miles (see Figure 7). At the state and regional scales, these roads would blend together. Until we find an automated way to do it, we will manually displace one or all roads affected on each map product.

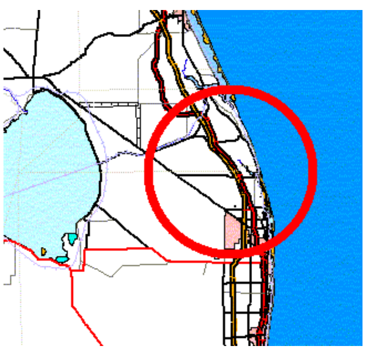

Figure 7. Convergence of Interstate 95 and Florida Turnpike. This shows actual positions of roads without generalization.

Figure 8. Convergence - generalized on State map product.

Figure 9. Convergence - generalized on Regional map product.

- Barrier Islands. Some coastal areas of the United States have barrier islands located just offshore. Examples of these include the Florida Keys, the Outer Banks, the Jersey shore, etc. These islands are connected to each other and to the mainland by bridges and causeways. At a state or regional scale, some of the islands are so narrow that they disappear under the thickness of the highway symbol. The smaller islands have to be enlarged and displaced along with the roads that traverse them, so that they can remain visible. The examples below are from the Miami area.

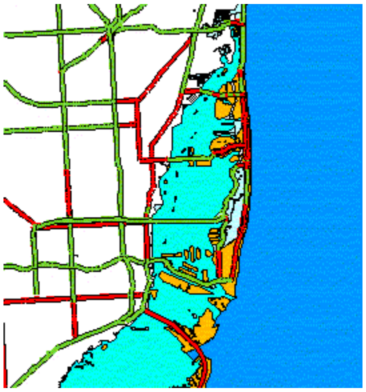

Figure 10. Offshore islands in Miami area without generalization. At this scale, roads cover up the smaller islands.

Figure 11. Generalized roads and islands on the State map product.

Figure 12. Generalized roads and islands on the Regional map product. At this scale islands are very abstract.

- City Tints. Our state and regional map products show major city boundaries in yellow tint. The city boundaries sometimes need to be enlarged for the yellow tint to be visible at smaller scales. See Figure 13.
- City Line weights. Since major roads converge on cities, AAA has historically "knocked down" the symbology of roads to a thinner line weight within the city tint boundaries. This reduces clutter, and allows the roads to be displayed more clearly. See Figure 14.

Figure 13. City Tints enlarged for visibility on regional map product.

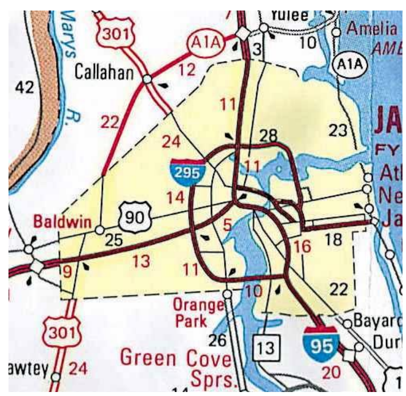

Figure 14. City line weights knocked down in urban areas. This example is for the Jacksonville area.

Updating the Database and Map Products

At AAA, our products are on regular update cycles. This means that all the work that we do for a map this year, will have to be revisited when the map comes up for update next year. In the meantime, our master database has been changing; underlying vendor data has been swapped out, and AAA proprietary information added. When the time comes to update the map, we would like to preserve all the effort that went into making the map the first time around, while also taking advantage of all the changes that have occurred in the database since the map was first created.

One approach would be to simply generate the map again. Perform all the generalization steps again on the new database to derive a new map. Who cares what the old map looked like - just create it again with current data. If map production were fully automated, this would be an acceptable course of action. However, the map production system is not fully automated. There are a number of manual, interactive steps involved in producing a map. The output of our Automated Extraction needs to be reviewed and adjusted by the cartographer. A considerable amount of research in invested in making sure that the map includes the appropriate features, that they are properly located and labeled. The index needs to be checked and verified. Since a lot of manual effort is put into map production, we would like to preserve this effort if possible, and not have to recreate it year after year.

The other approach would be to have a system of notifying the cartographer of any information which could influence the next revision of a map. The cartographer could then retrieve the previous version of the map, examine the changes recommended by the system, make any adjustments necessary, and be done with the map. The advantage of this approach is that all generalization work is preserved and does not need to be done again.

Update is an issue at the fringe of the generalization puzzle. If I had a system that could generalize a map perfectly every time, I would probably not care so much about map update - I would simply re-generate the map with new data. But since we don't currently have a fully automated map generalization system, I do want to preserve the manual, interactive effort that went into making the map in the first place, so I don't have to re-do that effort every time the map is updated. We don't want to have to treat each revision as a brand new map.

Updating the Master Database

AAA updates the master database throughout the year as new information becomes available. In addition, we refresh the master database with new vendor releases. We have the challenge of loading new vendor data while at the same time preserving AAA modifications to the database. From one vendor release to another, features may be deleted from the database, added to it, or (as in the case of realignments) removed and re-added to the database.

The fact that our database is so dynamic does have an impact on how we update our map products. For example, the map I create this year will have to be updated next year. Over the course of a year, the underlying database will have changed. The problem is then, how to bring the appropriate changes that have occurred to the master database down to the product layers, bringing in the new information while at the same time preserving product-specific changes and generalization.

We are trying to deal with database update and map update as separate issues. In other words, the problem of how to preserve master level changes from one vendor release to another is different than the challenge of how to apply changes in the master database to particular map products. In this view, database update has nothing to do with the maps that were derived from the data. From this perspective, it is easy to see map update as a separate issue. We have a good workflow in place to update the database with new vendor data, while at the same time preserving AAA value-added information. We feel we have the database update problem resolved, and are now ready to focus more attention on the map update process.

Updating Map Products - preserve previous version's work while accessing latest information

Map update is not a problem on the top of everyone's mind when it comes to generalization, but it is a real need for a publisher like AAA who must update map titles every year. We find that a lot of organizations typically make one-time maps, rather than maps to be maintained year after year. Organizations that do maintain maps generally do so at the individual map level, not at the master database level like we are attempting to do.

The relationship between generalization and map update may seem like a bit of a stretch, but it is a very real need for us. It could be argued that it is not a generalization problem at all, but we would like to see continued research in the GIS community regarding how generalization and map update should work together.

Ideally, when it came time to update a map, the system would make any necessary additions and deletions for us. Under our current model of creating product coverages or layers, it would be useful if an automated generalization process could look for new features and remove or replace old features automatically based on Permanent ID, or by comparing last year's generalization to this year's.

Until then, we have to handle the Map Update process manually. Our current approach to map update is described below, in the hopes that future breakthroughs in the automation of generalization will make this kind of interactive work unnecessary.

Re-extract features

Features in the master database may have changed since the map was last produced. In order to get the latest map information, all features for the map are extracted again to local coverages. Automated Extraction is run again to get a new default set of layers for this map type and scale.

- Find new features in the database that should go on the map.
- Find features on the map that have been deleted from the database, determine if they should be removed from map.

Change Detection - Compare old product layers with new extracts

Once the layers are re-extracted, the cartographer can compare the new extracts with the old map. We can provide some automated assistance, such as identifying all Permanent Ids that are new, or all PIDs that appear in the new map but no longer appear in the new data (deleted or realigned). These reports can give the cartographer a set of old and new (or changed) features to examine, but it is still up to the individual cartographer to determine if a feature should indeed be removed from or added to the map.

- Identify changed features, such as a realigned highways. In this case, the old features need to be dropped from the map and the new features added from the master database.
- Preserve cartographic displacements. If a feature is displaced for cartographic reasons, preserve that displacement, even for features that are replaced or realigned.
- Preserve other forms of generalization. If features are simplified, or typified, and they need to be replaced, the same generalization operations will have to be performed again on the replacement features.

Performance of UNIX Workstation Arcinfo and NT Desktop Arcinfo

A lot of our ideas and plans for generalization are based on our experience with UNIX Arcinfo. With the Tourbook Map project, we have now split our development and research efforts between two platforms (UNIX and NT). We continue to use UNIX for sheet maps and heavy duty data crunching, but have made a philosophical decision to limit our enhancement of the UNIX map production system while we learn what ArcMap can do. We are waiting to make the switch to the desktop. Performance is the factor that will cause us to switch to the desktop, or stay with UNIX.

We are just starting to work with the desktop. We know we can make maps with ArcMap once the data layers are prepared through the UNIX Automated Extraction. We have yet to get into geodatabases and modeling generalization behavior into our objects. It is also unclear to us how much generalization technology is currently available through ArcGIS, or how advances will be phased into future product releases.

As we become more comfortable with Arcinfo 8.1 desktop, and as the performance of this product improves, we may be able to do Automated Extraction, and all generalization, on the desktop.

We look forward to working with the object-oriented technology of the ArcGIS desktop. This is new technology for us, and we are particularly interested in the following capabilities:

- Building behavior into an object so that it would know how to represent itself in different products and at different scales

- Building sophisticated behavior that would consider the relationships between different objects in selecting which objects should appear on a map.
- Performance of these operations in a real-time production environment
- Can the object-oriented approach facilitate the map update process ? Smart objects that automatically update themselves in map products.
- Is ArcMap capable of producing one of our very large sheet maps ? Is there a size limitation for acceptable performance ?

Will processing speed and software ever get to the point where maps can be generalized and customized in real time, or will GIS software developers write tools to automatically propagate master level changes to affected map products ? Or will we have smart maps that update themselves ? This conference is an excellent opportunity for the users and vendors in GIS to compare notes on what is important to them. The user community needs to let GIS developers know how they are using their software, what we would like for them to improve, as well as what features are not as important to us.

Miguel Garriga

Project Manager

GIS Development & Support

1000 AAA Drive, mailstop 56

Heathrow, FL 32746

mgarriga@national.aaa.com

Geoff Baldwin

Senior Cartographer

GIS Cartography

1000 AAA Drive, mailstop 56

Heathrow, FL 32746

gbaldwin@national.aaa.com
