ESRI UC 2003 Paper 700

Egads, They're Making Maps!
Controlling Chaos with a Tracking Extension

Charlie Richman, GIS Manager, District of Columbia Office of Planning

The good news is that rolling out ArcView to the masses means that everyone can make maps. The bad news is, they often do. There are several approaches for providing standard map templates and data loaders. Still, core GIS staff face requests like, "I need to update the map that XXX made for me last year–the one with all the blue on it." How can you ensure that casual users keep track of their maps in a useful way? We describe our experience customizing ArcGIS to "help" (force) users to document all of their maps in a central database.

Background.

Making maps is fun, and is easier than ever with modern GIS tools. It's not nearly as much fun to file and organize information about the maps you've made. This essential conflict is a problem even for small GIS groups, but when GIS tools are rolled out to the masses it can be a nightmare.

Typical results include (1) complete chaos, where no one knows who made what maps when, (2) tight handcuffs that limit the ability of general users to make ad-hoc maps, (3) folder-based filing systems that sometimes help, but often fall short, or (4) full document management system implementations, which can require substantial changes in office practice. None of these seem good solutions for offices that want to disseminate powerful, general-purpose tools to a broad audience.

The District of Columbia Office of Planning (OP) illustrates this pattern perfectly. In our ArcView 3.x era, a dedicated cadre of GIS experts followed reasonably well-structured rules for managing GIS projects and products. They built simple tools including a data loader and standard map templates for their own use and for a broad spectrum of planners across the Office. The predictable result: numerous maps that varied in quality, and numerous time-consuming searches for old maps -- like "can you help me find that map that color-coded neighborhoods? I know I've seen it. I think someone who used to work here did it a year or two ago" (The author was interrupted by that exact query while writing the previous paragraph!)

Based on experience in OP and elsewhere, we elected to make automatic map tracking part of our base implementation for ArcGIS 8.x

Our Office.

Our planning office now has approximately 70 staff, including half a dozen full-time GIS specialists. Most planners have at least some familiarity with ESRI GIS products and

have had ArcView 3.x on their desks for some time. Some planners rely on the existing ArcView 3.x extensions we delivered to generate fairly standard maps; others use them as jumping off points for truly creative presentations and analysis. The GIS staff provide support in depth to planners in our office and to sister agencies, and also provide maps and data to the public. Our office serves diverse needs from development review and historic preservation to neighborhood planning, comprehensive planning, and long-range planning. It is a high-visibility agency, and responds to multiple rapid-fire requests from internal and external customers on an ongoing basis.

Our PCs are powerful enough to run ArcGIS on each desktop, and we have floating ArcGIS licensing to support our users. We store our information on central servers, either as flat files or in SDE. We have in-house staff with experience developing tools with Visual Basic 6, and we chose that platform for our initial ArcGIS tool deployment.

Our Approach.

We did not expect users to take the time to document which .mxd made any particular map, where it was saved, or what it represented. And we were not confident that users would voluntarily tag each map with a unique tracking number we could look up in the future when the inevitable requests for reprints or revisions arrived. So we designed a framework that does some of this for them, and "encourages" them to do the rest.

We created a relatively simple ArcMap extension called OpTools, and are deploying it to each ArcGIS desktop. Its core functionality is packaged in a single ActiveX DLL created with Visual Basic 6. It does not implement the iExtensionConfig interface, so users cannot decide whether or not to load it. It loads every time ArcMap is started.

When the ArcGIS application framework loads, it triggers the extension startup event. OpTools compares its own time-and-date stamp to the copy of optools.dll it sees stored on a central server. If the server has a newer version, we spawn a separate process to run an installer program and perform an update; ArcMap can then be terminated. Once the installer recognizes that ArcMap has indeed terminated, the .dll is updated. This means that the software is self-updating, and new functions can be deployed without revisiting each desktop. Users do not need administrative rights on their desktops to perform these updates, only to perform the initial installation of OpTools.

When OpTools loads, it automatically stores user information in a central tracking database. We track ArcGIS utilization by user, by license type, and by session duration. This helps enormously in monitoring license usage, and helps us understand who is and isn't using ArcMap on a regular basis. It helps us understand who our "power users" are, and who needs encouragement (or training) to make better use of GIS tools.

Once a document is loaded into the ArcMap environment, OpTools inspects the main toolbar and main menu, replacing key items with our own substitutes. The most important items are the print button and print menu item. We also add buttons for our own custom functions to ArcMap. Our Print substitutes call the original print functions – but only after we've launched our own tracking dialog. Users simply can't print or export a map without going through that dialog. They can't make any maps that we

haven't recorded and indexed, and every map our office makes will contain a valid tracking number.

Any time a user attempts to print (or export) a map, we check to see if there is a text element on the layout that contains a tracking number; if so, we look up the information we've already stored for that tracking number, and present it to the user for review. We ask if the map is a reprint, a minor edit, or significantly different than the previously printed map. Based on their response, we either update the information for that tracking number or generate a new one. We record the Windows UserID and the date and time of printing for that map in our tracking database, together with the path and filename of the .mxd that made the map. (If the .mxd is as yet untitled, we address that issue later when the user attempts to close the document.)

For a new or significantly modified map, we require users to provide a short description and a series of keywords. We assign it a unique tracking number, record the information in the central database, and put the tracking number on the layout ourselves through code. This ensures that tracking numbers actually appear on all maps when they are printed. For data frame image exports, we temporarily add the tracking number as a graphic element on top of the data frame, perform the export, and then remove it again. Either way, the tracking information we need will be on every map our office makes.

The resulting tracking database contains records of every user who's used ArcMap, what licenses they used and for how long, what maps they made, what .mxds were associated with those maps, and tracking numbers for each map. This means that we can now search for maps by user, by date, by keyword or description, or by the tracking numbers printed directly on them.

In addition to tracking functions, OpTools provides a suite of tools that automate many standard tasks. These include: a standard map tool that allows users to create commonly requested maps in a variety of layouts; a robust data loader that allows users to browse data sets alphabetically or thematically, or through searching by keyword while displaying metadata; a "map series" tool that produces sets of related maps based on an initial layout; and functions that streamline the use of raster catalogs. We expect OpTools to continue to grow for some time.

Our Experience.

OP's GIS staff have used OpTools throughout the development period, receiving incremental updates as new builds are posted to the central server. Beta testing by a larger group of users has begun, and we're planning general rollout and training for the office as a whole.

No system is perfect, and we've learned some lessons along the way. Developers typically have more privileges than typical users, and it's easy to create errors if database permissions are not set to allow all users to update their tracking information in the tracking database. We need to take care to maintain binary compatibility over

time so our bootstrap installer can update OpTools.dll without needing to run regsvr32 or regedit, both of which require administrative rights that typical users may not have.

We discovered issues with early versions of the Compile and Register add-in for VB at version 8.3; these only reinforced our reliance on SourceSafe, a server-based source code management solution from Microsoft. SourceSafe ensures that multiple developers can work in parallel without interfering with each other, and that we can roll back changes if modules are accidentally corrupted.

We've had to make compromises in forcing users to enter information. For example, we'd prefer to force them to save their .mxd before printing so we always have the name of the corresponding .mxd when we're recording tracking data. However, many of our maps are really standard maps or slight variations on them – and we don't want to be any more intrusive than necessary. Instead, we store which of our standard maps was used to manufacture a printed map if the underlying .mxd has not been saved. (When the user closes the document later, we do point out that they printed it and ask if the .mxd ought to be saved – and if they do save it we keep that information.)

Mostly, we've come to recognize that even a simple mandatory dialog when printing can be irritating – because we're the ones who are faced with it the most. It's helped us understand the need to streamline it as much as possible. But we're more convinced than ever that it's an essential tool for our planning office to help track our own work. The irritation of entering basic information for each map you make is far less of an annoyance than re-making a prior map simply because we couldn't locate the earlier work.

Acknowledgments

Dennis Waardenburg was the lead developer for much of this work, and helped to review this manuscript.

Author Information

Charlie Richman
GIS Manager
District of Columbia Office of Planning
801 North Capitol Street NE, Washington DC 20002
voice: 202 442 7621
fax: 202 442 7638
charlie.richman@dc.gov