

# Supporting Interactive Editing of Cartographic Representations in GIS Software

**Cory Eicher & Marc-Olivier Briat**

ESRI, Redlands, California, USA  
ceicher@esri.com, mbriat@esri.com

## ABSTRACT

Though cartographic production based on geographic information systems has proven benefits, a major obstacle to the wide acceptance of GIS as a tool for production cartography has been its inflexibility in allowing cartographers to make changes to the representations of geographic features on a feature by feature basis. Consequently, many cartographers work either solely in a vector commercial graphics software package, or they divide their work between GIS and graphics software packages.

Software is being developed at ESRI based on the geodatabase to support map finishing in ArcGIS. Cartographer-centric tools allow users to modify feature symbolization and geometry. These tools work with the geodatabase and store changes as overrides that do not effect the underlying GIS data. This paper describes the cartographic finishing software under development at ESRI, with a focus on the design and function of manual cartographic editing tools, supplemented by tools for automating common processes.

## 1 INTRODUCTION

This paper addresses the challenge of how to develop a map finishing software system that is integrated into a geographic information system. In this context, a GIS is defined to include: the GIS application software, the underlying relational spatial databases, and also the analysis and automated workflows associated with the working system. Such a system needs to meet the requirements of cartographic users – cartographers and cartographic editors – while also serving the needs of GIS and mapping system administrators. Traditionally, GIS-based mapping solutions have not met cartographic user requirements, however a new software system is proposed that targets these needs by providing cartographic users increased flexibility and productivity. Also, because cartographic finishing facilities are integrated with GIS, the system continues to leverage the advantages of GIS for data compilation, management, and analysis. Furthermore, adding cartographic data and functionality to a GIS realizes the benefit of managing GIS and cartographic data, workflows, and processes, within a common system.

A new map finishing software system is currently under development at ESRI as part of the ArcGIS software product. The system allows for the storage of feature symbolization – or *representation* information – together with vector feature data in ESRI's geodatabase. The system also provide for exceptions – or *overrides* – to the database-stored representation rules. Representations and overrides constitute the basic building blocks of a powerful system for creating and storing symbolization in a geodatabase. This paper focuses on the advantages of this system for interactive manual cartography supplemented by automated GIS processes.

The remainder of this paper is organized into major sections covering: user requirements for map finishing software, an introduction to the ArcGIS map finishing software under development at ESRI, descriptions of the system concepts of representations and overrides, and a discussion on how overrides can be used to support different map symbolization needs. Finally, before concluding, some information is presented about the specific manual cartographic editing tools that will be provided in the system.

## 2 USER REQUIREMENTS OF A MAP FINISHING SOFTWARE SYSTEM

### 2.1 Flexibility

Despite advances in the cartographic facilities of GIS software [Murad-al-shaikh 2005], many cartographers still prefer to work outside of a GIS, in a graphic software package or CAD system. Similarly, many organizations use their enterprise GIS to perform geographic data compilation, management, and analysis, but use a separate, usually non-GIS system for map finishing. One reason for this is because GIS enforces a strict structure on how vector feature data are symbolized.

Classically, feature symbolization is created in a GIS by assigning a set of symbol elements (e.g. markers, pen strokes, fill patterns) to features based on one or more GIS feature attribute values. Because a feature is assigned its symbol based on its attributes, to change its appearance – or representation – one must change its attributes. Changing the appearance of a particular feature on a particular map can be a difficult or impossible process in existing GIS mapping systems. In many cases cartographic editors do not have permission to change feature attributes that drive symbology. In other cases, depending on the symbolization requirement, enough unique cases can exist that the cartographic editor needs to create a new symbol element for each symbolized feature. Most importantly, cartographers have difficulty in not being able to reposition or otherwise modify the geometry of features for the purposes of the current map without adversely affecting the use of this data for other purposes including: analysis, maintenance of topological integrity, or other mapping or analysis functions.

Vector graphic and CAD software packages offer cartographers the flexibility to interact with map feature representations as graphics. Also, since editors are working with graphics and not GIS data, they can freely modify graphic geometries. Finally, most packages offer editors the increased flexibility to work with either “wholes” or “parts” of representations. Of course, freedom doesn't come without cost. Though graphic systems do provide some organizing functionality, e.g. graphic layers, a more organized system is often desirable for managing and updating map content. Additionally, the fact that graphics aren't GIS data can create additional challenges to keep GIS and map graphic data synchronized.

A GIS-based system that provides (where needed) the flexibility of a graphic-based mapping system, could allow cartographers and mapping organizations to finish their maps inside the same system where the data is compiled, managed, and updated.

## **2.2 Productivity**

A second reason why many mapping organizations choose to use vector graphic-based software packages for map production is the availability of interactive editing tools in these packages. These tools allow cartographers to be very productive in their manual work. It would be a stretch to describe these tools as easy to use, especially for novice users, but many experienced cartographers swear by them. After mastering a short list of seemingly basic (but beneath the surface very powerful) geometry selection and editing tools, many tasks can be performed quickly and with ease.

To woo cartographers away from simple graphic software systems, a set of familiar, and equally productive tools must be provided inside the GIS software. The tools currently provided in most GIS packages for manipulating geometries are designed for the compilation and management of geographic data. New tools need to be made available to work with cartographic data. This holds true for manipulating geometry, and also for modifying the appearance of symbolized features. Furthermore, to be adopted by cartographers, these tools need to have behaviors and graphical user interfaces that are similar to the tools present in graphic-based systems.

In addition to achieving increased efficiency in manual work, cartographers, cartographic editors, and the managers of GIS-based mapping systems benefit from increased productivity by automating much of the map finishing work that is still performed today through tedious, manual processes. There are many possibilities for improved automation of the map finishing process: Just having basic GIS query and facilities available can improve productivity, as can automatic geoprocessing of data for compilation as well as for data/map quality checking. Tools are being developed at ESRI to geoprocess cartographic representations to flag and then review areas of representation conflict. These tools are natively aware of symbolization, and therefore are able to detect visual conflicts that can exist even where the underlying geometries are cleanly separated (Fig. 1).

Additionally, new representation functionality is also becoming available that automates the drawing of dashed or dotted lines, ensuring a “good fit” for the pattern at the start and end points of every symbolized line (Fig. 2). This frees the cartographer from the arduous task of correcting poorly rendered lines, and allows them to move on to tasks more suitable for human creativity.

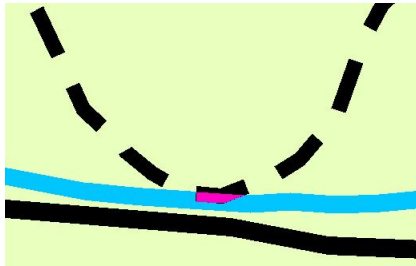


Fig. 1 – Representation Conflict Detection (pink area is a conflict between black dashed road and blue river)



Fig. 2 - Dashed Lines with Automatic Fit-to-Length

### 3 REPRESENTATIONS AND OVERRIDES SYSTEM OVERVIEW

#### 3.1 ArcGIS and ESRI's geodatabase

Cartographic finishing functionality is being developed at ESRI for a future release of ArcGIS [Hardy and Kressmann 2005]. This functionality includes database stored *representations* – to store symbolization information, and support for *overrides* – which allow exceptions to the symbolization rules. Integrating this map finishing functionality with an existing commercial GIS software package provides many advantages (more listed above), including the opportunity to leverage existing GIS facilities for analytical mapping, and the convenience of managing geographic data updating and cartographic finishing in the same system.

The mapping functionality being developed for ArcGIS is heavily centered on relational database technology, specifically the ESRI geographic database, or geodatabase. The geodatabase is a scalable relational database that supports the storage of feature and raster datasets, topologies, relationship classes, specialized network datasets, and now representations. Single-user, file-based geodatabases can be created, and enterprise geodatabases can also be built on top of widely used commercial database technologies including: Oracle, Microsoft's SQL Server, DB2, and Informix. ESRI's ArcSDE technology is used to access enterprise geodatabases, and this technology supports versioning and multi-user editing [ESRI 2004]. Additionally, the geodatabase and ArcSDE form the basis for future functionality that supports the incremental updating of cartographic representations [Briat et al 2005].

#### 3.2 Feature class representations

Two new data objects are added to the ESRI geodatabase to support the storage of feature symbolization information: A *feature class representation* stores symbolization information for a set of vector features in an existing feature class, and a *representation dataset* is a container for a set of feature class representations.

A feature class representation consists of one or more *representation rules*, and each feature in the feature class is assigned a single *representation rule*. For example, in a roads feature class, one might create a feature class representation of those roads, with four representation rules for: highways, arterials, local streets, and gravel roads.

Feature class representations are the essential component of the new mapping system, and they offer three key benefits:

- Feature class representations offer the advantage of storing symbolization information in a geodatabase, as opposed to keeping this information in binary project/map files or individual data layer symbolization files. Many organizations, particularly those with enterprise geodatabases, have asked for the ability to manage symbolization or representation information together with their geographic data in a geodatabase.
- Feature class representations define feature symbolization information in a more flexible manner from how symbolization is defined for regular GIS features. Feature class representation symbolization is built up from a system of basic (and fast drawing) symbols: markers, strokes, and fills. Common map representations can be created using only these basic symbols, and more advanced representations can be created by introducing *geometric effects* that preprocess geometry before drawing. For instance, using the above roads example, local streets can be represented with a basic stroke symbol, which defines width and color, whereas for gravel roads a Dash geometric effect can be inserted to dynamically produce a dash pattern that is then drawn by the basic symbol (Fig. 2). Geometric effects can be chained to produce more advanced effects. Importantly, both basic symbol properties and the properties of geometric effects (e.g. the dash pattern template) are defined in representation rules, and also available to editors to *override*. Overrides are covered in depth in Section 4.

- Because several feature class representations can be defined for the same vector feature class, the technology supports the definition of multiple representations of the same geographic data in the database with the geographic data. For example, a representation of roads might be created for a street atlas, and a different representation created for a topographic map for hiking.

### 3.3 Representation datasets

*Representation datasets* are simply containers for feature class representations. Feature class representations are dependent on feature classes, which can be organized into feature datasets. Representation datasets fill a similar role for feature class representations. It will be common to maintain a separate representation dataset for each different map product being produced. In the multiple representation case, two different representations of the same feature class – e.g. street atlas roads and topographic hiking roads – can be organized into different representation datasets – e.g. street atlas representations and topographic hiking representations (Fig. 3).

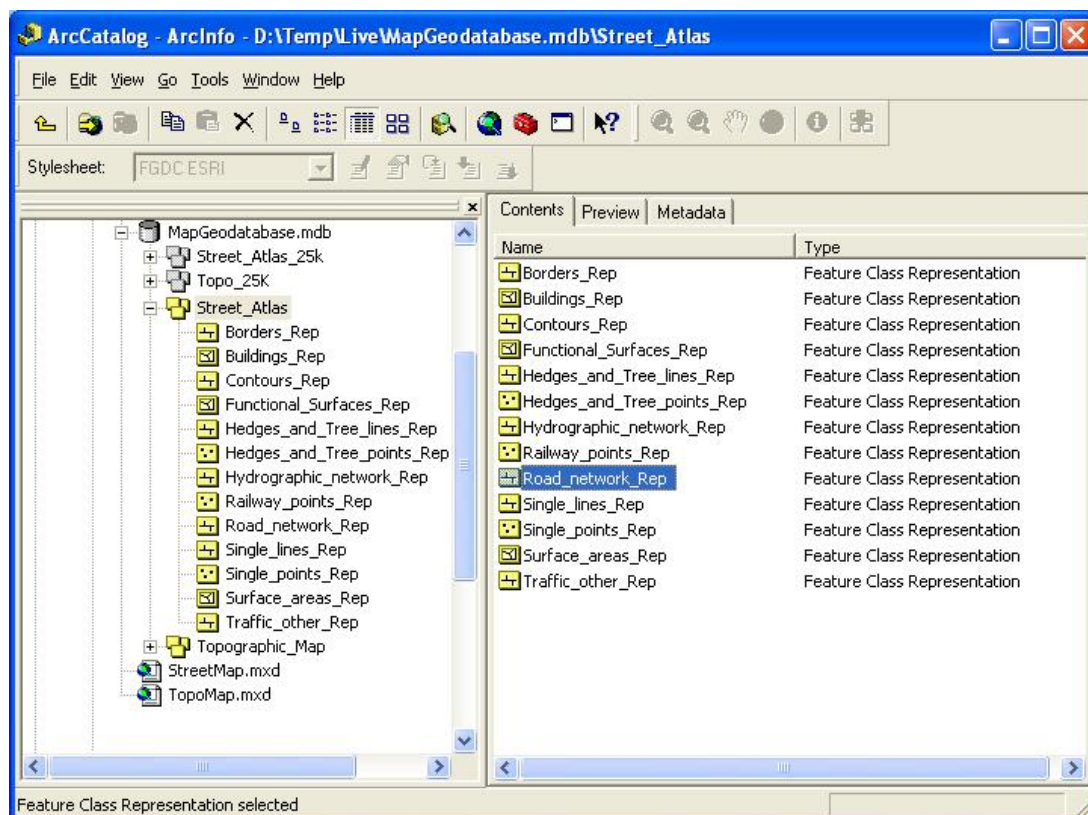


Fig. 3 - ArcCatalog view of a Geodatabase Containing Representations

### 3.4 Support for overrides

An *override* is a per-feature exception to the representation rule defined in a feature class representation, and overrides offer some compelling possibilities in a mapping system built on feature class representations. They can be used to store exceptions created by manual editors, and also to create powerful linkages between explicit feature attribute fields and map representations. Overrides are covered in depth in Section 4.

### 3.5 Intuitive graphic editing tools

Under development for the new system are a set of manual, interactive cartographic editing tools designed to edit feature class representations. An analysis of user requirements has shown that the basic tools should work similarly to tools available in commercial graphic and CAD software packages. Tools are provided for working with representation geometries and graphical properties. These tools are covered in depth in Section 5.

### 3.6 ArcGIS geoprocessing for automation

Most maps cannot be produced to an exacting standard without some manual intervention, and so interactive cartographic editing tools are necessary in a good map finishing system. However, methods of automation are equally important to reduce the amount of manual work. ArcGIS 9.0 introduced a flexible geoprocessing framework and set of geoprocessing tools for this purpose. The framework can be used to automate data compilation, analysis, and other workflows. The system has several options for user interaction including: dialogs with forms and buttons, a scripting interface supporting Python and other scripting languages, and a visual modeling environment where multiple tools and scripts can be linked together on a canvas, allowing you to create higher order models to automate larger or more complex processes.

The existing ArcGIS geoprocessing system is extended for cartography by adding geoprocessing tools that work with cartographic representations and overrides [Hardy and Kressmann 2005]. New tools are provided to create and manage representations. Also, importantly, general purpose data compilation and analysis tools can be used to support cartography in the system, including chaining these together to produce higher order cartographic models.

### 3.7 Customization options for developers

Although the cartographic facilities being discussed are being added to ArcGIS, which is a commercial off the shelf software product, it is worth mentioning that the system is very customizable for developers. The world has many mapping organizations, and therefore very many map specifications. Even within the European continent, topographic map specifications can vary widely from country to country. In fact, the map specification is often a source of pride for each national mapping agency, and for good reason as the map look of a map often represents centuries of national cartographic history and decades of producing electronic map product data.

The ArcGIS map finishing system allows developers to create custom components to meet the specific symbolization requirements of any organization. The system is built on ArcObjects and can be customized with many object-oriented development languages including VB, VB.NET, C++, and Java. Additionally, at a higher architectural level the mapping system is designed to be easily extended. For example if a developer writes a custom geometric effect using ArcObjects, the effect just plugs into the system, automatically taking advantage of overrides and working with the out-of-the-box interactive representation editing tools and automated geoprocessing functions.

## 4 OVERRIDES

### 4.1 Concept

*Representation rules* stored as part of feature class representations provide *structure* in the form of rules for symbolizing feature data, whereas *overrides* provide *flexibility* to be able to symbolize an individual feature differently from the representation rule.

Like representation rules, overrides are stored in the database as part of the feature class representation, but unlike the rules, which apply to a set of features, overrides operate per feature. Overrides can include exceptions to basic symbol properties, as well as the properties of geometric effects. An even more powerful capability is to use overrides to store alternate representation geometries for features, this without changing the geometry of underlying features. Multiple overrides for a given feature can be stored together in a default field, or explicit fields can be defined in the underlying feature class to separately store overrides for different properties. Finally, a free representations allows you to completely override a feature representation, giving you flexibility to work with the feature's symbolization as an arbitrary graphic.

### 4.2 Overriding graphic properties

When finishing a map, it is often necessary to modify the appearance of an individual feature, and representation overrides make this possible. A cartographer can change, for example the color of a particular marker, or the line width of a particular feature. Or, perhaps more realistically, he or she could choose to hide a particular feature on a particular map. This is possible as visibility is an overrideable property available for all representations.

As a more advanced application of overrides, where representation rules are defined with geometric effects, editors can modify the properties of these effects, again on a per feature basis. So, for example where an Offset effect is in place for line feature representations, a cartographic editor could choose a different offset distance (or even no offset distance), for a particular feature or set of features. As an other example, a set of polygon representations may be represented

according to a rule with their outlines dynamically simplified by a vertex thinning algorithm. In this case, an override can allow different polygons features to be simplified using different thinning tolerances. These tolerances are stored as overrides to the representation rule.

### 4.3 Overriding Geometry

As well as graphical properties, a cartographer can override representation geometries for individual features. For example, in a representation of polygon building features, her or she can override the position of a particular building's representation without affecting the underlying source geometry. As another example, for a set of a linear road feature representations, they can reshape the line representation geometries, turning some straight segments into Bezier curves and reshaping these curves to avoid visual conflicts with other map features. Again, this can be done without changing the actual shape of the roads as they are stored in the feature class. This allows other database users to continue to connect to the unmodified source features for analysis, routing, or other purposes.

The capability to override representation geometry also allows the cartographer to flag some line vertices as special control points to constrain the phasing of dash and marker patterns. Interactive tools are provided to create/remove these vertices manually, and geoprocessing tools are also available to automate this process. For example, one might run a geoprocessing tool to flag vertices along a line where there are sharp changes in direction. Then, when the line pattern is drawn, solid dashes be centered on each control point vertex, and the pattern will automatically be stretched to accommodate this constraint (Fig. 4 and Fig. 5).

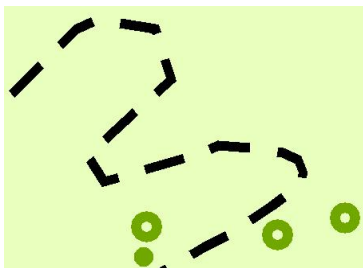


Fig. 4 – Dashed Lines

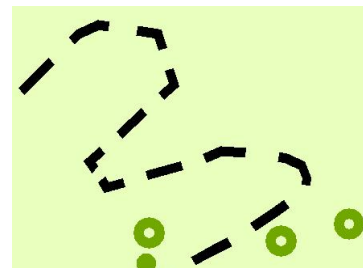


Fig. 5 - Dashed Lines with Representation Control Points

Where different representation geometries are required for the same features in different maps, multiple feature class representations can be created for the same feature class. This allows editors to use geometry overrides to change the representation geometry in one map, without affecting the representation in another.

### 4.4 Modeling overrides

The override mechanism provided by the proposed mapping system is generic. Quite simply, feature representations are drawn based on their assigned representation rule, unless there is an override for a particular property for a particular feature, in which case the override value is used. So, for example, the representation rule for a set of building representations might be to draw the buildings in black, but for one or more buildings, overrides may exist which changes the color of the drawn building from black to red.

Though the mechanism for overrides is generic, there are two ways in which overrides can be stored: First, one can choose to store all overrides in a default override field, which is a Blob or binary large object field. If one creates a new feature class representation and overrides some properties by editing in ArcMap, these overrides will be stored in the default override field. The second option for the user is to store overrides in an explicit field. For example, for line representations, one might choose to store overrides to the line width in a floating point field. One might also, for the same representations, choose to store overrides to the visibility property in an integer field (where 0 is interpreted as False and non-zero values are interpreted as True). This would allow a simple SQL query to find all invisible features.

### 4.5 Using the default override field to store manual edits as exceptions

The intended use for storing overrides in the default override field is to provide a mechanism for the cartographer or cartographic editor to modify individual representations and to store these changes as exceptions to the representation rule. These changes could include geometry overrides. Using the default Blob override field for this is convenient as it is provided when you create a feature class representation, and overrides are automatically stored here unless otherwise specified. The default override field is a good place to store overrides created through manual editing, especially if the user doesn't have any particular reason to structure their storage.



## 4.6 Using values stored in explicit fields to generate and update map symbology

The system also provides the ability to specify that overrides of particular properties be stored in specific fields. The primary intended use is somewhat the opposite of using the default override Blob field. While the default field is good for recording manual exceptions as representations are edited, explicit fields that are *already populated* can be used to "drive" the map drawing from the database. Field values may already exist in the field, for example a populated integer angle field might already exist in the representation's feature class table and could be used to orient a set of point representations. Alternatively, a new field could be added and then populated by running a geoprocessing tool that is specifically designed to populate the field with useful values for cartography. For example, a *cul-de-sac* cartographic geoprocessing tool is provided in the system to calculate an integer Cap field for a set of line features. Values are calculated, based on whether or not a line is a dead end, and then the Cap field can be mapped to the Line Cap property of the line representations. The result is that so-called *cul-de-sac* roads are symbolized with a square cap – the override value, while all the others are drawn with round caps – the representation rule value.

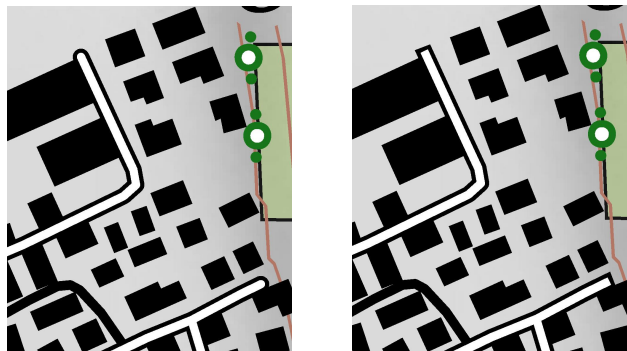


Fig. 6 – Explicit Field Override used to Override Line Cap Property (Before and After)

Regardless of whether or not a field is pre-calculated, or calculated by running a cartographic geoprocessing tool, the main purpose of explicit field overrides is to use values that already exist in the data to generate a more finished map.

It is also possible to use explicit field overrides for a purpose similar to the described use for the default override field. One can map a property to an explicit field *that is initially empty*, so as to record manual changes made to a particular property (e.g. line width) for particular features. By storing these changes in an explicit field, it is easier to track changes and update data using standard GIS database functionalities.

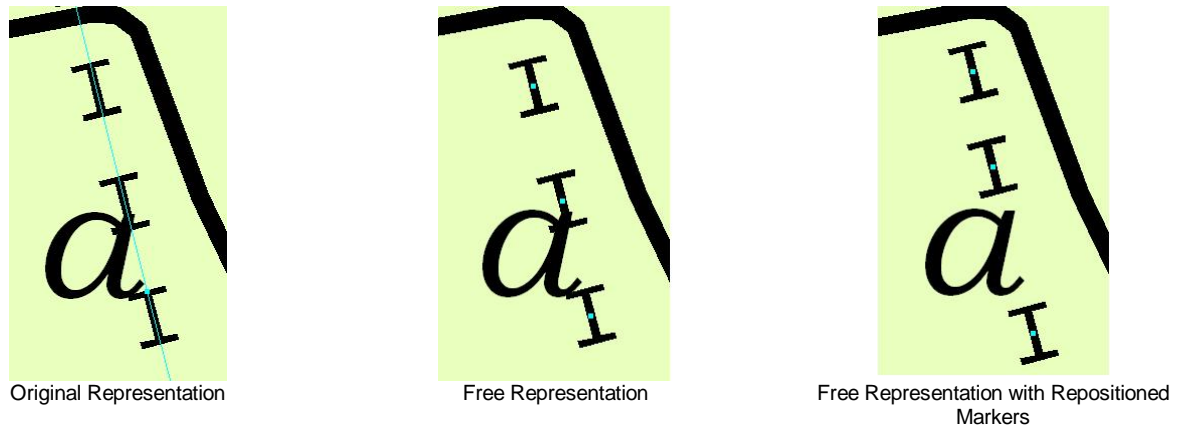
## 4.7 Free representations

*Free representations* are the most extreme form of override. With a free representation the entire representation is overridden. That is, a complete copy of the geometry and its representation rule is stored in the default override field for a particular feature. In this condition the feature representation is in the most flexible state possible in the system. Because a free representation no longer references a shared representation rule from the representation feature class, the cartographic editor is free to make changes that are not possible when overriding individual properties.

There are several intended uses of free representations:

- Use a free representation to be able to change the graphic structure of a particular feature's representation. For instance, with a free representation one can change the basic symbol type of an individual feature representation. For example, by rule city areas may be represented as polygons, but free representations allow an editor to symbolize several instead with a marker at the center of the area.
- Use free representations to be able to draw additional graphic shapes that are associated with the feature. For example, a railroad yard might appear on a map with multiple spurs and several polygonal platforms, however in the GIS data the entire yard might consist solely of a polygon that just represents the geographic extent of the feature. The cartographer could convert the polygon to a free representation and then create new line and polygon graphics to artistically represent the rail yard based on the map context, personal knowledge, or some other information unavailable in the geographic database.

- Use free representations to get very detailed control over the graphic marks that constitute a representation. For example, a linear border feature might be represented by a repeating pattern of 'I'-shaped markers. The editor can convert this representation into a free representation, thus decoupling it from the shared list of representation rules. At this point, he or she can reposition the individual markers to avoid conflict with other map features (Fig. 7).



*Fig. 7 - Using a Free Representation to Reposition Individual Markers along a Line Representation*

It should be mentioned that the system has been designed for users to make light use of overrides, especially free representations. This is because maps created with many overrides and free representations begin to duplicate data and can create inefficient performance as a result. Administrators must consider the use of the system for producing a particular map product. If many overrides or free representations are envisioned, the system design should be modified, most likely by adding additional representation rules that can be referenced from multiple features.

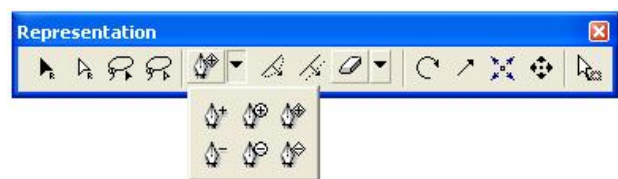
## 5 REPRESENTATION EDITING TOOLS

### 5.1 Purpose

To complete the map finishing system – on top of the structures provided to store feature representations in the database, and built in override capabilities – a set of manual editing tools are provided for map finishing (Fig. 8). All the tools are designed to work on cartographic representations. The tools can access basic symbol properties, the parameters of any geometric effects, as well as representation geometry. Also, editors use the same set of tools to work with free representations as for editing rule-based representations.



*Fig. 8 - Representation Editing Tools*



*Fig.9 - Vertex Editing Tools*

### 5.2 Basic selection and geometry modification tools

Two basic interactive representation selection and geometry modification tools are provided to the user (Fig. 8, left side of toolbar). The first (black) tool works on whole feature representations. Editors can use this tool to select and then change representation properties (e.g. fill color, marker size, etc.). The tool also allows the editor to interactively rotate or resize one or more selected representations.

A second (white) tool is provided that works on the underlying representation geometry. The tool is able to work with parts of one or more representations. With the tool users can select portions of linear or polygonal geometry, reshaping this geometry by repositioning the selected vertices.

In most situations, when repositioning or reshaping representation geometry, the system provides 'what you see is what you get', or WYSIWYG, visual feedback. This can be convenient for repositioning representations with respect to other representations. Snapping is also available for this purpose.



### 5.3 Changing graphic properties

A graphic properties editor window is provided that shows the properties of selected representations (Fig. 10). Where appropriate, when multiple representations are selected, common properties are shown so that representation properties can be updated en-masse. Changing a representation property using the graphic properties window creates an override, and the window also indicates with an icon any property that is currently overridden for the selected representation or representations.

It should be mentioned that because overrides are stored as values in feature class tables, the general task of editing a representation (for example changing the size of a representation marker, or reshaping the geometry of a line representation) works in much the same way that regular feature attributes and geometries are edited in ArcGIS. Representation editing takes place within a feature edit session, edits are immediately visible on the map, and edits are not saved to the database until the session's edits are saved. Additionally, when representation data are stored in an ArcSDE enterprise geodatabase, edits can be saved to a version of the data and then later reconciled and posted to a parent version. ArcSDE versioning supports multi-user editing and a long transaction model for updating representation data.

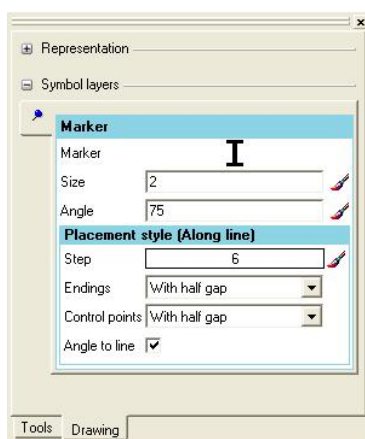


Fig.10 - Graphic Properties Window Showing a Rule-Based Representation with Three Overrides

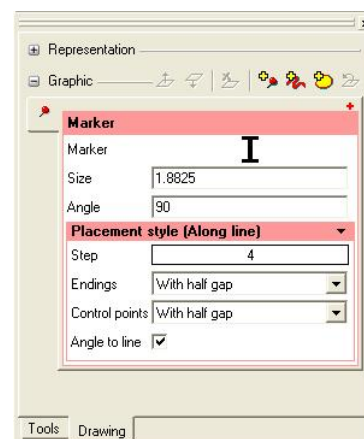


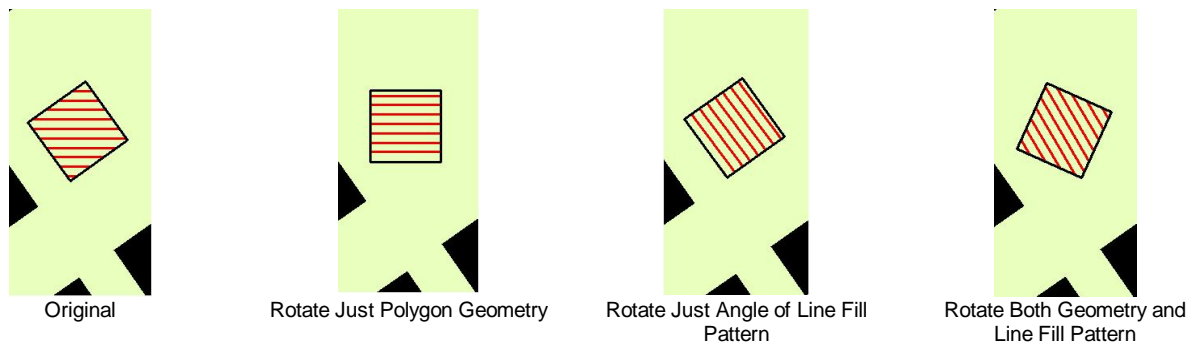
Fig.11 - Graphic Properties Window Showing a Free Representation

### 5.4 Vertex editing tools

Cartographers commonly manipulate linear geometries and polygon outlines manually. A set of helpful tools is provided to work specifically with vertices along these linear geometries (Fig. 9). Tools are provided to quickly add or remove a vertex, to change a vertex to a Bezier control point (or the reverse), and to flag a vertex as a representation control point (or not). A Bezier control point defines that a connecting segment is a curved segment, as opposed to a straight line. A representation control point specifies that a vertex be treated as a special point in the phasing of some representation effects. For example, dashes can be drawn along a linear geometry and constrained to control points (Fig. 5).

### 5.5 Tools for interacting with geometry or graphic properties

A set of specialized representation editing tools is provided that allows users to interact with geometry or graphic properties. When activated, each of these tools presents a list of graphic properties that can be modified by the tool, as well as geometry. The user chooses which properties to edit, including geometry, then the tool is used to interactively change the selected properties. For example, a user might select a polygon building representation, and then use the Rotate tool to rotate either the polygon geometry, the line fill pattern, or both (Fig. 12).



*Fig. 12 - Using the Rotate Tool*

## 5.6 Editing free representations

As previously described, free representations allow editors additional freedoms for editing feature representations. No special tools are needed to work with free representations. The user simply needs to select one or representations, and run an explicit operation to convert to a free representation. From here the user can work with the free representation using all of the previously described representation editing tools. When a free representation is selected, the graphic properties window changes its appearance (Fig. 11). For free representations a pink color is used for section headings, and also a set of buttons is enabled allowing a user to add, remove, and reorder the representation's basic symbol components.

## 6 CONCLUSION

A map finishing system has been presented, based on the concept of storing feature representations in a geodatabase with the capability to override representation rules on a per feature basis. The system provides much of the flexibility of performing cartographic finishing work in a graphic software system, but this flexibility can be employed as needed. A majority of GIS features can be represented based on symbolization information stored with the features in a geodatabase, while database-stored overrides can be used where exceptions to the rules are necessary. Overrides allow exceptions to the graphic properties defined in representation rules, as well as to representation geometries. Free representations offer an additional level of flexibility. Finally, a set of interactive cartographic editing tools is provided that work on both rule-based representations and free representations. The resulting system presents intriguing possibilities for large-scale mapping organizations, for smaller cartographic production shops, and for those that just want to produce a better map using GIS software.

## 7 NOTES

1. The sample data used in Fig. 1, 2, 4, 5, 6, 7, 12 is swisstopo VECTOR25, copyright Swiss Federal Office of Topography.
2. Special thanks to the members of the ESRI team working on representation and overrides, particularly to Paul Hardy from ESRI for his careful review of the paper.
3. This paper is a forward-looking document, and the capabilities it describes are still under development. As such, it is intended to give guidance as to likely future direction and should not be interpreted as a commitment by ESRI to provide precise capabilities in specific releases.

## 8 REFERENCES

- Briat M, Monnot J, Kressmann T, 2005, "Incremental Update of Cartographic Data in a Versioned Environment", 22nd ICA Conference Proceedings, A Coruña, Spain
- ESRI 2004, ESRI White Paper, "Versioning workflows", <http://support.esri.com/index.cfm?fa=knowledgebase.whitepapers.viewPaper&PID=19&MetaID=722>
- Hardy P & Kressmann K. 2005. "Cartography, Database and GIS: Not Enemies, but Allies!", 22nd ICA Conference Proceedings, A Coruña, Spain
- Murad-al-shaikh M, 2005, "Professional Labeling and Annotation Techniques with ArcMap" 22nd ICA Conference Proceedings, A Coruña, Spain

## **BIOGRAPHY**

Mr. Eicher has worked at Environmental Systems Research Institute (ESRI), Inc. since 1999 as a Cartographic Software Specialist. He received a B.S. in Civil Engineering from the University of Virginia in 1997 and an M.S. in Geography from the Pennsylvania State University in 1999. While at ESRI Mr. Eicher has been involved with the design, implementation, testing, and documentation of cartographic software. He is a member of the Association of American Geographers and the current Non-Academic Director of the Cartography Specialty Group of the AAG. Mr. Eicher has been an active participant and presenter at meetings of the Association of American Geographers, the North American Cartographic Information Society, and the International Cartographic Association.