An Incremental Geographic Update System (IGUS) for a Large Geographic Database in New York City

J. Ding[1], S. Ahearn[1], E. Cooper[2]
[1] CARSI Hunter College, 695 Park Avenue New York, New York 10021
[2] Terraprise Inc. 3205 Magazine Street New Orleans, LA 70115

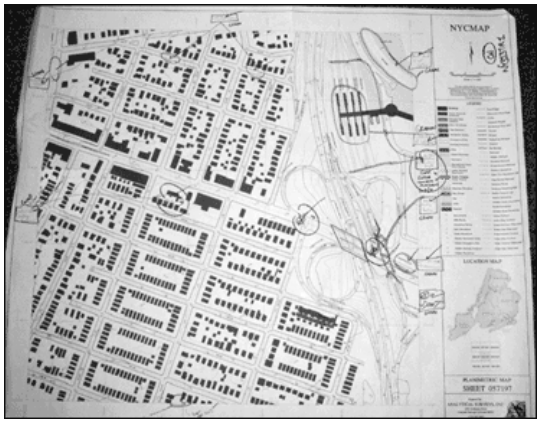
ABSTRACT

Over the last decade a number of states and cities in America have successfully established spatial data infrastructures that incorporate core spatial data such as the physical infrastructure layers, topography, road and rail networks and critical administrative databases like land parcel and administrative boundaries. New York City built its digital photogrammetric base map (NYCMap) from 1996 1:4,500 aerial photography and successfully integrated it with the City's existing land parcel file (COGIS) and street centerline file (LION) in the year 2000. NYCMap is composed of over two-dozen planimetric features (e.g. buildings, street centerline, towers, subway lines, railroad, hydrography and topography, etc.) and a 0.25-meter resolution digital orthophotos. It is one of the core spatial data sets widely used by City Government Agencies, decision makers, planners, lawyers and utility companies. A key issue facing the City of New York was what was the best methodology for updating NYCMap and how could it be done in such a way as to maintain the integrity of the database and track changes over time. This paper discusses a solution to the update of NYCMap with the creation of an Incremental Geographic Update System (IGUS).

IGUS establishes the techniques for the incremental maintenance of large spatial information system in three fundamental ways. First, because objected oriented GIS technology represents the geographic entities as objects instead of layers or geometries, changes can be tracked for each entity over time rather than changes of layers over time, which doesn't make sense from a temporal standpoint. Second, the approach combines the quality control and data integration process into one step and only operates on the changed features. This increases efficiency of GIS database maintenance and ensures data integrity. Third, incremental changes can be propagated to other database environments through the use of difference streams between database alternatives created on different dates. These changes can then be propagated out to other users in a generic data transfer format.

INTRODUCTION

Providing the most updated and accurate geographic information is one of the challenges faced by the GIS industry. GIS software architecture needs to be able to handle very large spatial databases and resolve conflicts among different versions of updates and it also needs to support update and validation of spatial data while preserving spatial data integrity. The traditional way to update large geographic databases is by wholesale replacement of an existing database with photogrammetric updates that incorporates the changes into the old database. Problems with this approach are twofold: 1) the data integrity is compromised as most photogrammetric systems don't enforce topology rules and validate changes and 2) Quality assurance of the geographic data is conducted on the entire database using geographic tiles with errors being transmitted via paper printout of these map tiles (Figure 1). Another problem is the difficulty of propagating database changes to other users in various GIS software environments. Propagation of the changes is usually only executed once the entire database has been updated, which can take up to two years for large urban regions.



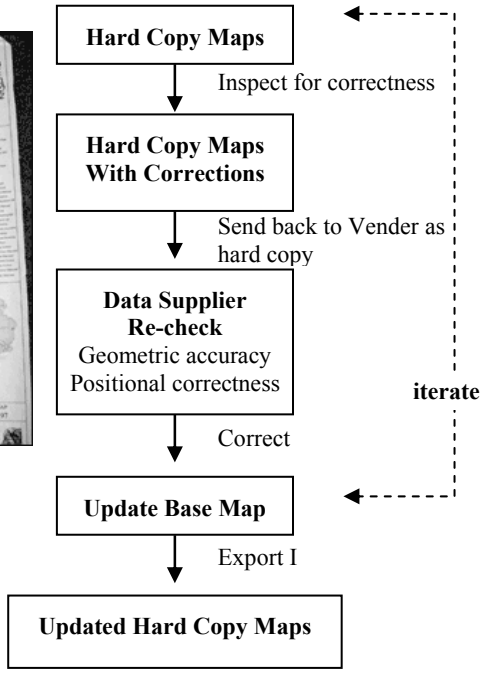


Figure 1: Traditional way to update large geographic databases

This research develops and formalizes an Incremental Geographic Update System (IGUS) that solves the problems discussed above using object-orientation and database versioning techniques, which are particular valuable in handling large spatial databases. The system has several distinguishing characteristics: 1) all geographic features have unique and persistent identifiers, 2) feature changes which are either updates, insertions or deletions are sent by the photogrammatrists in a generic ASCII format, 3) change features are brought into a single quality assurance (QA) object which can have geometric characteristics of point, line or polygon, 4) QA objects are reviewed and accepted or rejected, 5) accepted objects are integrated into the database, 6) only the rejected objects are sent back to the photogrammatrists with error remarks for redrafting. The process is repeated for multiple rounds until all QA objects are accepted. Incremental propagation of new geographic features is managed using daily checkpoints and by running difference streams between the database date of the client's last update and the current date. The resulting geographic features are then sent to the client (NYC agencies) in a generic ASCII file or in GML. Because all features have persistent and unique identifiers change management for the City Agencies is made possible.

Quality assurance process for the first Photogrammetric Basemap for New York City (NYCMap: 1996) was accomplished by dividing the city into 1800 0.65 km^2 tiles and running a set of automated procedures for geometric, topologic and attribute validation and a set of manual procedures that checked the accuracy of features capture. All feature capture errors were recorded on paper printout of tiles and sent back to the photogrammetric compliers for correction (Figure 1). This was an extremely time intensive process that was seen as being the biggest hurdle to update efficiency. When new flights were commissioned for 2001-02 a redesign of the update process was made. The redesign used the version management capabilities of the GE Smallworld database for organizing and overseeing the update process and for providing a mechanism for change management of features between subsequent updates to City Agencies. At the core of the redesign efforts was an incremental update process that used object-oriented technology to create a "Quality Assurance Object" (QA Objects) that could manage all geographic objects whether they were points, lines or polygons. This Incremental Geographic Update System (IGUS) was used to integrate changes in features that occurred between 1996 and 2002 into the 150 GB NYCMap

database. A new set of "super tiles" were created that were 2km by 2km in size. Tile boundaries were fuzzy in that an object that spanned more than one tile was assigned to a single tile. For massive features like "open space" (i.e. parks and cemeteries) objects were updated in a single citywide data delivery group. There were dozens of other design issues that needed to be reconciled for the successful implementation of the IGUS system, some of which will be discussed below. As a result 2001-02 update process over 250,000 features were inserted, deleted and modified in the NYCMap database.

THE CONCEPTUAL MODEL FOR INCREMENTAL UPDATE

The objectives of a conceptual model for an incremental geographic update system are to create a better and more efficient way to represent entities on the ground, enhance data integrity and to trace the behavior and changes of objects over time. Two technologies have made this achievable: object-oriented systems and version managed data stores. The advantage of object technology is that it more closely models real world objects than geometry centric data structures (Egenhofer and Frank 1989, Henderson and Ahearn) and reduce the impedance associated with the modeling process (Worboys, 1994). It also allows users to organize spatial information in an object-oriented manner that captures essential components of spatial temporal behavior of objects (Ahearn et al. 2001). Version managed data stores (Newall and Esterfeild, 1990) enable multiple users to update a geographic database, with each update process having its own logical copy of the complete dataset. The conflict among the versions of update can be resolved by selecting the right version update. All of these aspects make IGUS feasible and much more efficient compared to layer-based spatial databases.

Object transactions

The development environment that we are using is the GE Smallworld version 3.2, which is a version-managed object-oriented GIS. The NYCMap database contains over 30 feature tables. There are four types of geometry for the real world objects in the base map system: Point, Chain, Simple Area and Complex Area (multi-polygons, holes). In the GE Smallworld system an object can be represented with one or more of these geometry types. Each object table, for each feature group (i.e. transporations), contains records with a unique and persistent ID (UPI). The Incremental Geographic Update System (IGUS) uses the UPI to manage objects in all aspects of the incremental update process. UPI serves as an identifier for the feature in both supplier's and user's database and is the key for transactions. The UPI also makes it relatively easy to trace the history of a feature. A block of unused numbers for each object type is reserved for the future new inserted objects.

There are basically three kinds of changes that have occurred to features on the ground over time. They are "add a new object", "delete a existing object", and "modify the existing objects". Each of these changes is stored in a "QA/QC object" that can store point, line or area features in the same data object table.

Data representation and data transfer format specifications

Data representation and data transfer format is a key issue in the incremental update process. A generic ASCII data transfer format of real world objects was developed to facilitate the transfer of the changes of spatial data between data user and data supplier who may have differing platforms. The incremental changes can be propagated to other database environment through the use of difference streams between database alternatives created on different dates. These changes can then be propagated out to other client database in the same generic data transfer format.

IMPLEMENTATION OF INCREMENTAL UPDATE

The operational process for incremental update is as show below (Figure 2). The data supplier only captures the changed objects between the previous photogrammetric compilation and the current one. Change objects are imported into the QA/QC object tables and the changed objects are comparing with new orthophotos and other relevant data associated with basemap by an operator. A QA/QC object will be integrated into the system automatically once it passes the visual check. Rejected objects will be returned to the supplier as digital objects in the generic ASCII format, with a reference to the problem. After the all updated objects are checked and accepted, a series of validation procedures is runs to ensure the consistency and connectivity of the objects with the system.

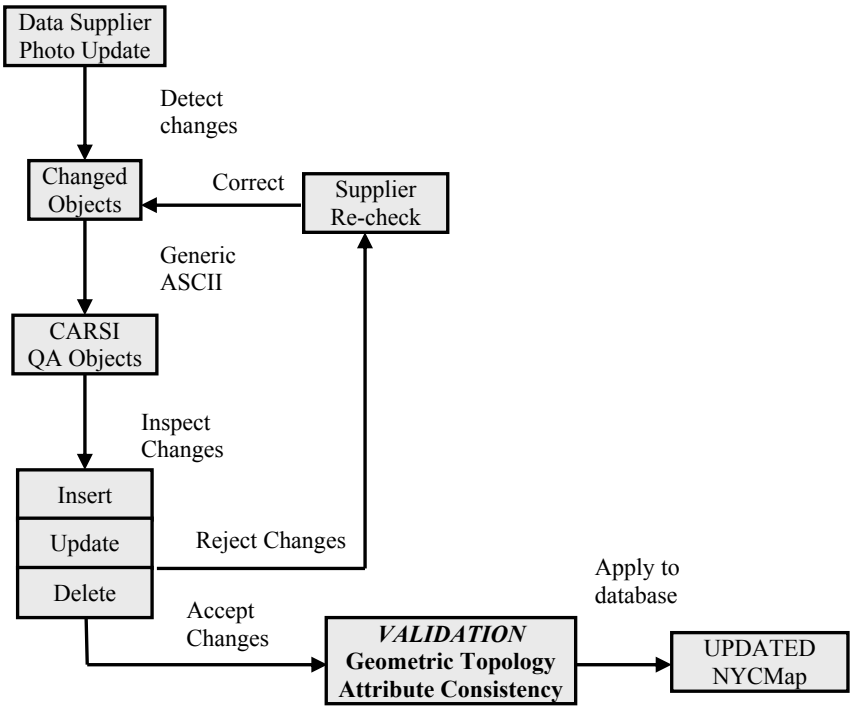


Figure 2: NYCmap Incremental update and database integration process)

RESULTS

The 1996- 2002 cycle updating process for NYCMap was conducted on approximately 320 tiles (2,000 m x 2,000 m). The average number of QA/QC objects, which represents the changes on the ground, was about 700. Average number of rejected QA/QC objects per tile is was 30. About 224,000 changes occurred in New York City from 1996 to 2001.

CONCLUSIONS

The work described above and experience gained by this research reinforced the view that objected oriented spatial systems are critical for the proper implementation of an incremental geographic update system. The unique and persistent identifiers are the key to delivering changes between data supplier and data user. OO support for multiple geometries, topological rules, validation tools and the object

transaction approach to update, preserve data integrity at the data receiver end and are key in the Quality Assurance process. This research also shows that the data transfer format developed in this research can be used both for transaction data update and propagation for changes between different software platforms. The incremental geographic update system for a large geographic database of New York City has proven to be a much more efficient and less expensive update system than traditional layer based update systems.

ACKNOWLEDGEMENTS

We would like to thank The Department of Information Technology and Telecommunications (DOITT) of New York City for funding this research. We would also like to thank GE Smallworld for their generous contributions to this effort.

BIBLIOGRAPHY

Ahearn, S. C.; J.L.D. Smith; A. R. Joshi; and J. Ding. 20001. "TIGMOD: an individual-based spatially explicit model for simulating tiger/human interaction in multiple use forests". Ecological Modeling 140 (2001) 81-97.

Eigenhofer, M. J. and Frank, A. U., 1989, Object-Oriented Modeling in GIS: Inheritance and propagation. In Proceedings of Auto-Carto 9, Bethesda: ACSM & ASPRS, pp. 588-598.

Henderson, D. B., S. Ahearn, 1998. Evolution of a municipal landbase from layers to objects, URISA Annual Conference Proceedings, Charlotte, NC.,pp. 320-330.

Newell, R. G. and M. Easterfield, 1990. Smallworld GIS: version management-the problem of the long transaction. Smallworld Technical Paper 4, pp 9.

Worboys, M. F. 1994. Object-oriented approaches to geo-referenced information. International Journal of Geographic Information Systems, 8: 385-399.