Rethinking Gazetteers and Interoperability

Greg Janée
Institute for Computational Earth System Science
University of California at Santa Barbara
Santa Barbara, CA 93106-3060
gjanee@alexandria.ucsb.edu

December 9, 2006

1 ADL gazetteer protocol

The Alexandria project's desire to create a gazetteer protocol grew out of two needs. First, we had identified gazetteers as a key component of our architecture for distributed geospatial digital libraries, but having only one instance of a gazetteer to go by (ours), it was unclear which features digital libraries could generally expect from gazetteers, and which were artifacts of our implementation. Second, our initial work on gazetteers was largely content-based, having focused on developing the ADL Gazetteer Content Standard (GCS) [1], and a protocol would provide a complementary, functional definition of a gazetteer as a type of knowledge organization system.

Thus in 2001 we collaborated with ESRI to create the ADL Gazetteer Protocol [2]. In the interest of interoperability, the protocol defines a simplified gazetteer model that is compatible with the GCS, but omits much of the latter's elaborate details such as temporal qualification and lineage. In the simplified model, gazetteer entries are equated with conceptual places; each entry (i.e., place) is described by one or more names (notably, all names are unqualified), one or more spatial footprints, and zero or more feature types. Each of the aforementioned quantities may be flagged as current or historical, and one quantity within each category must be flagged as primary. The place as a whole may be flagged as current or historical. Named relationships may be asserted between places, though the relationships themselves are undefined by the protocol. The protocol defines just two services: download all entries, and search for entries that satisfy a boolean combination of constraints on place attributes.

Our experience with the protocol has generally been favorable. It fulfilled our original needs, and we and a few others have successfully implemented it on both the client and server side. However, several limitations have emerged, notably:

- *Lack of support for qualified placenames.* Searching for a place qualified by the name of a containing (and disambiguating) administrative unit is not directly supported by the protocol, and even assembling such searches out of the lower-level functionality the protocol *does* provide is onerous at best. This is a significant limitation given the ubiquity of such queries.
- *Conflicting and unpredictable semantics between spatial and relational searches.* The ADL protocol provides two ways of expressing containment constraints: searching spatially (e.g., find "Santa Barbara" spatially contained within California's footprint) and searching relationally (find "Santa Barbara" that has a PartOf relationship to California). Conceptually, these two types of queries are equivalent (or, at any rate, any difference between them is surely splitting semantic hairs), yet the results are likely to be entirely different due to implementation artifacts such as incompletenesses in the recording of

PartOf relationships, and problems testing spatial containment due to spatial footprints being overly small (points) or overly large (bounding boxes). Furthermore, these differences are, from the user's perspective, both unpredictable and variable from query to query.

The OGC's WFS gazetteer profile [3] is analogous to the ADL protocol. Though there are a number of significant differences, the two protocols are largely similar in their broad characteristics, and they target the same level of functionality. As a consequence, many of the statements regarding the ADL protocol below apply to OGC's as well.

2 Interoperability use cases

As time has passed since the initial development of the ADL protocol, and as mapping and geocoding services and "mashups" have become more common and easier to implement, we have found ourselves stepping back and asking questions such as: What does interoperability mean for gazetteers? What role can gazetteers play in some of these new services? Does it even make sense to speak of a *gazetteer* protocol? To begin to answer these questions, let's observe five use cases related to gazetteers.

Harvest Retrieve the entire contents of a gazetteer. Harvesting is necessary to support aggregation of gazetteers, particularly if we want to provide unified search over multiple, local, idiosyncratic gazetteers. Harvesting is also necessary to support certain intensive uses of gazetteers, such as in geoparsing, where any online protocol is bound to be too inefficient. Harvesting requires: a standard means of retrieving content, such as OAI-PMH [4], and if not a common representation of gazetteer content, at least a few well-known representations. A common feature typology would be helpful, but this is probably destined to remain a pipe dream.

Lookup Find a place by name or other description. This is classic gazetteer functionality, of course, but with this use case we're explicitly looking beyond gazetteers and including, for example, street address geocoding. Indeed, the trend in recent mapping services (Google, Yahoo, etc.) is to recognize more and more forms of spatial reference (airport codes is a good example), purely for user convenience. Note also that these services desire only simple point locations in return, in order to define a location of interest or a place to pan a map to; there is no use of extended placename information in this use case.

Reverse lookup More classic gazetteer functionality: find places near a given spatial location. More specifically, find places of a given type near a given spatial location. And even more specifically and possible more useful in many circumstances: find the *nearest* place of a given type to a given spatial location. As a hypothetical example of a use of the last type of query, consider a service that offers airport-related information. Users of the service would like to enter the most convenient (to them) form of spatial reference to identify an intended airport. Meanwhile, the developer of the service would like to take advantage of a lookup service that translates spatial references to airport codes, for the latter is how the service's information is likely to be indexed. A lookup service, such as described above, provides half the solution by returning latitude/longitude coordinates; a reverse lookup service can then be used to return the nearest airport.

Geoparse Identify and geocode the placenames in a document. This use case involves gazetteers, but is, strictly speaking, outside the scope of gazetteers for two reasons. First, geoparsing requires examining substantial context around candidate placename references, often the entire document. Second, a geoparser's use of a gazetteer is computationally intensive and specialized, and hence the gazetteer will need to be held close to the geoparser, say, in a local database. A nice example of a "mashup" service that combines geoparsing with another, widely-used protocol is the GeoNames.org RSS-to-GeoRSS converter [5, 6, 7], which automat-

ically geoparses and geocodes articles from any RSS feed, thereby allowing the articles to be displayed by a GeoRSS-capable map service such as the Acme GeoRSS Map Viewer [8].

Ontology Along the lines of the Semantic Web, we can think of a gazetteer as a (potentially distributed) kind of knowledge base of facts and associations that supports certain kinds of inferences. The key requirements here are that each concept (i.e., place) be uniquely identified (in RDF, this means by URI) and that associations be defined by an ontology. However, significant barriers remain to implementing this use case. If we are to work with more than one gazetteer at a time, then either there can be only one fact for each place, or unification of equivalent facts (i.e., places) is necessary; inferencing will fail otherwise.

3 ADL protocol, revisited

Returning to the ADL gazetteer protocol, let's look at how well it supports the above use cases.

Harvest The ADL protocol's simplified gazetteer model may provide a useful, "lightweight" counterpart to the GCS. Otherwise, the protocol's harvest service, lacking a restart or resume capability, is too simplistic to support large-scale downloads. In any case, OAI-PMH is well-established in this area.

Lookup ADL's lack of support for qualified placenames, and generally rigid query model, are too limiting.

Reverse lookup Reverse lookups are supported, but not nearest place queries.

Geoparse, Ontology These are outside the scope of the ADL protocol.

Thus it does not appear that a gazetteer protocol is, by itself, all that supportive of use cases that involve gazetteers. In the harvest use case, gazetteer interoperability centers more around representation; a general harvest protocol satisfies the functional needs. In the lookup use case, a protocol that is generalized beyond gazetteers is required. In the geoparse use case, gazetteers play a critical role, but not in an online sense; again, interoperability is centered around representation and harvesting. And likewise for the ontology use case.

4 Question for the workshop

As stated earlier, the ADL gazetteer protocol has proven useful, and will continue to be useful for accessing and defining gazetteer functionality. But in thinking of future development directions, should we rethink our approach to gazetteer interoperability?

In our past work we started with an entity (a gazetteer) and then defined a protocol giving access to that entity. Should we instead orient our efforts along the lines of the aforementioned use cases, and define protocols that are oriented around *functionality* (lookup, reverse lookup, etc.) as opposed to *entities*, protocols that various kinds of entities (gazetteers, geocoding services, etc.) can implement to varying degrees?

Simply refactoring functionality is not going to solve any difficult problems, of course: spatial search semantics will continue to be problematic and messy. But the suggested refactoring can provide value in a couple ways:

- The resulting protocols are likely to be more closely aligned with user needs and desires.
- The distinction between functionality (geocoding, say) and the entity implementing that functionality (a gazetteer or geocoding service or other type of entity) is clarified.

References

[1] Linda L. Hill (2004). Guide to the ADL Gazetteer Content Standard, version 3.2. `http://www.alexandria.ucsb.edu/gazetteer/ContentStandard/version3.2/GCS3.2-guide.htm`

[2] Greg Janée and Linda L. Hill (2001). The ADL Gazetteer Protocol.
http://www.alexandria.ucsb.edu/gazetteer/protocol/

[3] Jens Fitzke and Rob Atkinson, eds. (2006). Gazetteer Service — Application Profile of the Web Feature Service Implementation Specification. OGC 05-035r2, version 0.9.3.
http://portal.opengeospatial.org/files/?artifact_id=15529

[4] Carl Lagoze and Herbert Van de Sompel. The Open Archives Initiative Protocol for Metadata Harvesting.
http://www.openarchives.org/OAI/openarchivesprotocol.html

[5] Really Simply Syndication.
http://en.wikipedia.org/wiki/RSS_(file_format)

[6] Geographically Encoded Objects for RSS feeds.
http://www.georss.org/

[7] GeoNames RSS to GeoRSS Converter.
http://www.geonames.org/rss-to-georss-converter.html

[8] Jef Poskanzer. ACME GeoRSS Map Viewer.
http://www.acme.com/GeoRSS/about.html

Biography

Greg is a research specialist for the Institute for Computational Earth System Science and for the Map & Imagery Laboratory, Davidson Library, at the University of California at Santa Barbara. He is currently working in two areas: methods and architectures for discovery of distributed, heterogeneous geospatial data and, more generally, digital library support of Earth science data lifecycles; and long-term preservation of geospatial data. The latter work is for the National Geospatial Digital Archive.

Previously, Greg was technical leader of the Alexandria Digital Library Project, principal developer of the Alexandria Digital Library software, and developer of the ADL gazetteer and thesaurus protocols.

His experience prior to the Alexandria project was in software engineering for commercial and government clients in the areas of object-oriented class libraries; 2D, 3D and fractal-based visualization; rule-based expert systems; compilers and query languages; and embedded database systems.

Greg holds an M.S. in computer science and a B.S. (summa cum laude) in mathematics, both from the University of California at Santa Barbara.

Greg's publications and conference and workshop presentations are listed at http://www.alexandria.ucsb.edu/~gjanee/publications.html.