Database Design and the Electronic Publication of Archaeological Information

David Schloen
Oriental Institute, University of Chicago.

Computerized, database management systems are now very common in archaeology, as they are in other disciplines, that use large quantities of information. But many archaeologists know that even today, twenty years after the introduction of the first personal computers, we are far from realizing the potential benefits of information technology. This is because of the continuing lack of standardization in the structure of archaeological databases. Each archaeologist, understandably, wishes to customize his or her database system for the project at hand, but the resulting chaos, of different file formats and user interfaces, prevents easy electronic merging of detailed information from different projects. This hinders computer-based, archaeological research, conducted on wider, spatial and temporal scales. At the present time, the construction of comprehensive, regional databases requires either a central authority, that mandates a standard database structure for a given region, or, it requires much expense and labor, after the fact, to write special-purpose programs, to merge an assortment of disparate databases.

Neither of these two solutions is acceptable, in my view, except occasionally, and on a limited scale. Most archaeologists would agree that no prescriptive scheme, adopted at the outset, for recording the data from diverse excavation and survey projects, can or should be enforced, even within a single geographical region (see, e.g., Richards, 1985). Each investigator should be free to employ the terminology and the recording system, that is best suited to his or her project, and any computerized database should easily accommodate these preferences. As for merging differently structured databases, after they have been created, aside from the prohibitive effort and expense involved, the problem is that in the absence of a standard structure, the result will be, simply, another idiosyncratic database, which cannot easily be combined with still, more databases, on ever broader, spatial scales. Thus, the legitimate requirements of individual projects appear to conflict with the laudable ambition, long-held and widely shared among archaeologists, to create comprehensive computer databases for multi-site, archaeological comparison and analysis. That ambition has been stimulated, even more in recent years, by the explosive growth in the use of the Internet, which has prompted many archaeologists, who had previously paid little attention to database issues, to consider the feasibility of creating widely distributed, archaeological databases as media of publication and research. Yet, progress in this direction is limited by the natural tendency of individual projects, to rely on customized, database designs.

There is a way of resolving this conflict, however. Despite the inevitable variety of archaeological recording systems and terminology, there are basic features, common to all archaeological data, which permit a standardized database structure and a correspondingly uniform and intuitive user interface – although, at a more abstract level, than has usually been considered. Moreover, the standardization, I have in mind, does not prescribe the use of any particular terminology, or recording system. The requisite level of standardization can be achieved, simply by using what I will call, an "item-oriented" data model, instead of the "class-oriented" data model, that has been common in archaeology. *Class-oriented* databases typically provide one file, or database table, for each class or sub-class of archaeological observations (e.g., pottery, lithic material, metal objects, bones, botanical remains, debris layers, architectural features, entire sites, regions, etc.). In each file, there is a fixed number of fields, usually represented as table columns, which predetermine the attributes that are available to describe items of a given class. Each item is, therefore, represented as a row in a table, with a predefined structure.

Rigidly structured databases, of this sort, employ what has been called, a "strictly typed" data model (Tsichritzis and Lochovsky, 1982:8). As applied to archaeology, this means that decisions about the typology of archaeological observations (how many classes of observations will be considered and how many and what kind of attributes each class will possess) are all "hard-coded" into the structure of the database, from the beginning, and cannot be changed very easily afterwards. This "strictly typed", class-oriented approach is nearly universal, in archaeology today, in large part, because readily available, relational database management software tends to encourage it. Examples of such software include familiar names, like dBase, Paradox, and Access, for PC-based applications, and DB2 or Oracle, on larger computers. In the same category, we can also include Geographical Information System (GIS) software products like ARC/INFO and ArcView, which have become popular among archaeologists in recent years and include similar kinds of relational database features. To be sure, commercially available, data management software does not actually require a class-oriented data model, but in most business applications, and in the standard working practice that has been derived from them, relational tables tend to be equated rather rigidly with broad classes of data, in the manner I have described above.

Unfortunately, it is the rigidity of the prevailing, class-oriented data model, in which a predetermined set of attributes is prescribed, for each of a limited set of predetermined classes, which prevents the automated merging, or joint querying of data, from multiple projects, that employ different typologies for recording archaeological observations. An *item-oriented* database design, by contrast, makes this automated merging much easier. In an item-oriented database, the basic structural element is not the predefined class of items, but rather, the abstract archaeological "item", itself, as a unit of observation, with which any number of descriptive attributes may be associated. A "class" is, thus, not a fixed structural component of the database, but merely an *ad hoc* grouping of items, based on a particular set of query criteria. The building blocks of the database are the individual items themselves, whose specific attributes, the end-user defines, by linking each item to a potentially unique set of variable-value pairs. An item-oriented database can be easily adapted and extended, as needed by the end-user, without extra programming, by permitting the user to add new variables and new values to the pool of available attributes, and to rename or delete these, as necessary. Similarly, the description of an item can be changed, by attaching different variable-value pairs, without affecting any other items.

An important advantage, of the item-oriented data model, is that a clear separation is maintained, between the relatively, primary attribution of descriptive variables to potentially unique, individual items, on the one hand, and the multitude of possible secondary and overlapping classifications, of those items, defined according to investigators' changing interests and assumptions, on the other. This approach, therefore, respects the tremendous variability of archaeological data, because hundreds, if not thousands, of overlapping classifications, of the many items observed in any large excavation, are both possible and useful. But most important, when the time comes to merge, or jointly query, databases from separate projects, no special programming is required, and the only user intervention, that is needed, is to specify the equivalences among the sets of variables and values, originating from the different projects.

Somewhat similar kinds of item-oriented design have been advocated by other archaeologists (e.g., Andresen and Madsen 1992), but my own approach is much more radically item-oriented, with a simpler and more abstract structure, than has been used elsewhere. Also, I organize all of the individual items in the database into a hierarchical "tree" that represents the spatial containment relationships among the various units of observation. Of course, spatial hierarchy is only one possible view of the relationships among archaeological items, but it is, by far, the most comprehensive and inclusive view, in the sense that all archaeological observations can be located at some place in a spatial hierarchy. In addition, because a tree structure is self-replicating and has the same properties, recursively, at all levels, the spatial hierarchy of archaeological items is infinitely extensible, in both directions, both macroscopically and microscopically. This means that a tree-structured, item-oriented database can easily accommodate data from multiple sites and regions, on all spatial scales, and from both excavation and survey projects, using the same, simple design of independently linked items and attributes.

An unintended benefit of this recursive, item-oriented design is that, quite by accident, it matches exactly, the structure of XML, the new "Extensible Markup Language", whose specification was formally approved in February, 1998, as a standard, non-proprietary syntax, for representing richly structured data on the World Wide Web. An XML document is a tree of abstract, nested "elements", that possess "attributes" (equivalent to what I am calling, "items" and "variables"), which has been serialized into lines of tagged, Unicode text for easy delivery over the Internet. This means that archaeological databases, of the type I am advocating, can be very easily published on the Web as "distributed databases", in the form of XML documents, accessible from any XML-capable browser, running on any computer platform (e.g., Macintosh, Unix, or Windows). Such databases could then be jointly queried, in any combination, simultaneously drawing on a variety of different web servers, each of which would deliver a different sub-tree, that would be dynamically integrated into an overall spatial hierarchy, viewed as a whole by the user. By including suitable Web "stylesheets," expressed in Dynamic HTML, or the new XSL ("XML Style Language"), and adding some Java code to customize the Web interface, it will not be difficult to produce a cross-platform, distributed Web application for archaeological data management, that has most of the functionality of operating system-specific, database applications, built on the same data model (without having to distribute such applications, or having to use any particular operating system). Of course, XML-based representations of data can be generated from any database, using any data model, but the fact that the archaeological data model, I am advocating, matches the "native", recursive structure of XML, should facilitate its use for archaeological publication on the Web, and thereby, improve its chances of acceptance as a generic standard.

The main point to remember about both XML, as an abstract data representation standard, and my analogous, archaeological database design, is that they provide robust standardization, in terms of a basic *framework*, consisting of a tree of elements with their attributes, but they do not force any standardization in terms of *content*, and so, they supply all of the benefits of a non-proprietary, standard file format, without any restrictions on what you can do with it. That is why Microsoft, for example, has announced that its entire suite of applications (Word, Access, Excel, etc.) will use XML, as a common, Web-oriented data format. Moreover, this hierarchical, item-oriented data model is especially suitable for archaeological purposes, because: (1) the tree structure has an obvious archaeological interpretation, in terms of spatial containment, and (2) the flexible element-with-its-attributes data structure can capture the idiosyncrasies of highly variable archaeological data, in a way that class-oriented database systems cannot.

In summary, this item-oriented design permits both a high degree of individual customization, or *flexibility*, in recording archaeological data, and sufficient underlying standardization, or *rigor*, to make it easy to combine databases from different projects. Furthermore, because of its abstractness, simplicity, and generality, this data model provides not only flexibility and rigor, but also the open-ended *extensibility* of a self-replicating structure, so that a single database can include information from any number of excavated or surveyed sites, stored and retrieved in a consistent and easily understood manner. Thus, the two-fold purpose of the hierarchical, item-oriented data model, advocated here, is to facilitate not only project-specific, *data management*, but also *electronic publication* of archaeological data, in general (a kind of "publication", which in my view, should entail both seamless *data integration* and easy *data dissemination*).

I have implemented this data model in a full-featured, 32-bit, Microsoft Windows application, named INFRA, which is an acronym for "Integrated Facility for Research in Archaeology." It was developed, using Borland Delphi, version 3, and various third-party, object-oriented, software components (including ESRI's MapObjects ActiveX control), with data storage and retrieval, implemented via the Borland Database Engine, using interlinked relational tables to represent "items," "variables," "values," "classes," etc. The basic design was first conceived in 1989, and I have implemented it, since then, in increasingly powerful versions: first in DOS, then in Windows 3.1, and now in Windows 95/NT. In addition to a primary tree diagram interface showing the *spatial containment* relationships of database items, other diagrammatic interfaces are used to represent the *temporal sequence* of items and *spatial adjacency* relationships. These represent various, complementary "views" of the data, neatly encapsulating, in diagram form, the extrinsic relationships among units of archaeological observation, that are difficult to represent in conventional, class-oriented, database systems. Other "views" include a realistic *map* interface, that uses GIS techniques to manipulate mapping data as vector graphics (with the ability, also, to show a raster image as an underlay), and a hypertext, word-processing *document* view wherein an interpretation of the data can be written, that contains embedded links to item descriptions, photographs, plans, and so on.

I would like to emphasize, again, that this Windows software, and the data model that it implements, prescribes no rigid format or particular terminology for recording archaeological information. The archaeologist, who uses it, can create and label individual items, variables, and values, as needed, in a manner that is appropriate for the project at hand, and can associate these, with one another, in a variety of spatial and temporal (or "stratigraphic") configurations. Initially, this approach demands a higher degree of conceptual abstraction, yet it actually corresponds better to observed entities in the real world, which do not manifest themselves in the form of tidy classes of material, but as idiosyncratic, individual items. Moreover, the abstraction, entailed in working with a few simple, generic concepts such as "item," "variable," and "value," and with a few schematically represented spatial and temporal relationships, permits both flexible customization from the user's perspective and rigorous standardization, in terms of the underlying data structure. Most important, because of this standardization, the task of merging databases from different projects is quite easily accomplished by grafting in (via "cut-and-paste" or mouse "drag-drop" operations) the spatial tree of one INFRA database, as a new branch of the spatial tree of a second database, and then, defining equivalences (again via "drag-drop") between the two original sets of variables and values. The end-user is not forced to turn to a programmer to map one rigid and idiosyncratic table structure onto another, because the comparison between different datasets is done at a more basic level, between individual items and their attributes. And, the end result of such a merger is a more comprehensive database, which preserves the standard, underlying structure of a simple item-oriented hierarchy, but which also reflects the naming conventions and recording systems of the individual projects, whose data are incorporated within it.

The software that I have developed for the Microsoft Windows, operating system is just one implementation of the radically item-oriented design, I am advocating, although it demonstrates what I think is the best approach to representing both the *intrinsic* attributes and the *extrinsic* interrelationships of archaeological items in a standardized fashion. In its specific features, it also demonstrates the benefits of an item-oriented design, for integrating very tightly, within a single software application, an array of powerful, yet, easy-to-use functions, that have been tailored for archaeological use. These functions include: (1) raster image storage and compression (i.e., for scanned photographs); (2) GIS-style map viewing and spatial query features; (3) statistical aggregation of data, displayed in various sorts of graphs and tabulations; (4) editing of hypertext documents, that can contain numerous hyperlinks to database information; (5) the ability to execute complex queries, that employ both intrinsic attributes and extrinsic relationships among items, and to save the resulting lists of selected items as named "classes", that can serve as filters in the creation of reports, graphs, tables, and composite plans; (6) the ability to design *ad hoc* tabular views of the database and to save them as tables, or print them out, thus, supplementing a variety of built-in, standardized reports; and (7) the ability to import tabular data and convert it into an item-oriented hierarchy, and also to export various views of the data, for further analysis in a variety of file formats.

Most of all, it is the open-ended extensibility of the item-oriented and thoroughly generic, data model, which I have implemented in this Windows application, that makes this design especially suitable for the electronic publication of archaeological data (i.e., "publication" understood as multi-source *data integration* and *data dissemination*) on digital mass storage media, or on the Web. Electronic publication, although much touted for its speed and cost-effectiveness, will be of limited value in archaeology, unless its intended audience can easily view and analyze published data in full detail, using visual interfaces and complex queries, with the goal of testing investigators' interpretations and of combining data from disparate sources, to permit more broadly based retrieval and analysis. There is a long history in archaeology of creating localized, special-purpose datasets, in order to test specific hypotheses, or to construct particular models. But what is needed, to enhance future research, is a tool that will permit rapid, efficient, and open-ended "exploratory data analyses", on broader spatial and temporal scales. In this way, patterns in the data may be detected, that currently go unnoticed, and patterns that are found may be explored further with a speed and rigor, hitherto impossible. The achievement of such benefits is what makes the adoption of a flexible yet standardized data model so desirable.

References

ANDRESEN, J., & MADSEN, T. (1992), "Data Structures for Excavation Recording: A Case of Complex Information Management", in: LARSEN C. U. (ed.), *Sites & Monuments*, National Museum of Denmark, Copenhagen, pp. 49–67.

RICHARDS, J.D. (1985). "Standardising the Record", in: COOPER M.A. & RICHARDS J.D. (eds.), *Current Issues in Archaeological Computing*, BAR International Series 271, British Archaeological Reports, Oxford, pp. 93–102.

TSICHRITZIS, D.C., & LOCHOVSKY, F.H. (1982), *Data Models,* Prentice-Hall, Englewood Cliffs, New Jersey.