Torsten Madsen

Design considerations for an excavation recording system

Paper presented at the conference Design and use of field information systems. CAA-NL98, Ammersfoort 17/12 1998.

The paper appear as read with minor corrections to language and grammar. With one exception illustrations are not included.
Torsten Madsen 4-10-2003

In 1987 I got myself into trouble with colleagues in Denmark. At a meeting on rescue archaeology I stated, quote: “Research is a process of cognition resting on an interaction between theory and praxis. The excavation situation is the most important kind of practise within archaeology, and consequently the most important part of archaeological research does not take place behind a writing desk or in a magazine. It takes place during the excavation itself where the theory – the whole of the theoretical and empirical background of the field archaeologist – is considered and tested against the source-material being investigated” (Madsen 1988: 25). This statement led me to the conclusion, for which I was almost lynched, that all excavators - meaning those that are responsible for the day to day decisions in the field – should be liable to a full research evaluation based on university level standards. If they could not pass such an evaluation, they should not be allowed to excavate.

Behind this statement lay a growing concern with the nature of the heavily expanding “business” of rescue excavation. The recordings from the excavations were becoming less and less structured, and at the same time minimal. You were required to deliver a list of contexts, a list of finds, a list of drawings, a list of photos together with the finds, drawings and photos, and that was it. There was and is no requirement for a thorough interpretation of the findings or for a recording of reflections on observed interrelationships of the findings. Indeed, apart from the simplest interrelationships like object x was found in context y there are no explicit demands for recordings of interrelationships. Implicitly, it is assumed that you can see for yourself using the drawings and photos.

This type of recording practice where description deliberately is kept separate from interpretation is a direct outcome of a positivistic research model. Despite the fact that this kind of model is hardly defended by anyone in the archaeological community today it still prevails more or less unchallenged in the practice of excavation recordings (Madsen 1988: 24).

My concern with the practice of rescue excavations, and in particular the way they are documented led me into some lengthy discussions with my colleague Jens Andresen. Was it possible to create a formalised recording system capable of coping with the full complexity of excavation information and at the same time flexible enough to allow for widely differing recording practices? In 1990 at the Southampton CAA conference we presented our ideas, and eventually had them published a couple of years later (Andresen & Madsen 1992).

Basic to the solution we suggested was that we accepted the existence of the recording entities forming the lists of traditional recordings. That is: contexts; finds; drawings and photos. However, instead of trying to elaborate on these basic entities with fields for all kinds of information we would rely on the potential of relational databases keeping separate types of information in separate tables. Further, the complex web of interrelationships between individual recordings should not be recorded through fields of the tables themselves, but by the use of link-tables making it possible to set all kinds and any number of relationships between instances of all entities externally and internally. In

this way the user rather than the programmer would decide the structure of the recordings.

Also, and very importantly we added a new entity to be cross-linked with all the others. We termed it construct and its function was to hold interpretative statements related to all other entities. We described it as follows: “Constructs can be considered as statements of abstract properties ascribed to layers and objects. That is, apart from being an entity that relates layers and objects in various combinations, they also, by definition, add an inferential quality to the relation by way of a value label. You may think of the construct as a multi-dimensional cross-reference table with a label field, where you can add a value to the cross-reference. Any construct may refer to layers, objects and other constructs in any combination and assemble these to a unity with some implied meaning.” (Andresen & Madsen 1992: 55).

In 1993 we succeeded in getting money for a three-year project to implement our ideas. The system was eventually baptised IDEA standing for Integrated Database for Excavation Analysis. The project has now stopped, but a working system exists and has been tested. I shall return to our experiences with this system later, but first I will have to be a little more specific about our objectives.

It was very important for us to create a system that could be tailored by the end user. The five fully interrelated entity tables for contexts, finds, constructs, drawings and photographs, constituted the core, but we would not decide in advance how the linkage between the entity tables should be used. Should it be free or should it be constrained so that for instance a find can only be entered if linked to an existing context. To cope with this we implemented a data modelling capability, where you could decide how the basic entities should relate to each other in a particular excavation recording.

The next step was to establish ways in which the user could add qualities to the basic entities. What we did was to define types of qualifiers implemented through separate tables and linked to the basic entity tables. A simple example would be free texts or binary objects like graphics. More complex types of qualifiers were for instance events, archaeological dating, and classifications with associated descriptive variables.

Events consist of different types of actions, persons who are allowed to carry out these actions. In separate modules you define the actions appropriate for each basic entity and the persons who can perform these actions. During data entry the defined actions are available through forms, where you can select one or more actions, one or more persons allowed to act, and dates for carrying out the actions.

Likewise with the archaeological dating. Through a special module you could build up hierarchically organised dating systems as we normally use them in archaeology, and then through the data entry forms apply these to the basic entities.

The most complex but also the most important qualifier was the classification system with associated descriptive variables. There is good reason to take a closer look at our

objectives here, as it is closely related to the basic philosophy behind the IDEA project and our basic notions of what excavations are all about. Let me present you with a number of statements.

- It should be possible to classify any set of data through different classification systems at the same time. In principle the number of such classification systems applied simultaneously to any particular set of data should be unlimited.
- It should be possible to add and change classification systems "on the fly" while being used for recording. However, the integrity of already recorded data should be enforced (i.e. it should not be possible to delete description or classification elements, if they are already in use).
- It should be possible to incorporate tree-structures in the definition of the classification systems and use the logic of these structures to operate on classified data, not least in search operations.
- It should be possible to categorise and quantify all relations between recorded data.
- It should be possible to assign an unlimited number of description variables to each class.
- Inheritance between classes should be established so that all variables of a class automatically exist as variables of all its subclasses.
- All definitions of classification and description systems used must be kept in the database, so that all recorded data becomes understandable without the use of external information.

Classifications can be applied to finds, contexts and constructs to the degree of specialisation and detailing needed during an excavation, fitted specifically to the type of object excavated, and continuously adjusted to the precise nature of the findings. You may have some standard of classification you wish to apply to all excavations, but that does not mean that you should not be able to apply other classifications or indeed create new classifications during the course of excavation.

There is one overriding reason why we need a dynamic approach to classification in relation to excavation recording. Increasingly, it has become clear that the research process can be viewed as a dialectic process between a theoretical modelling on the one hand, where pictures of the past are created, and a data modelling on the other, where observations are organised into meaningful structures (Madsen 1995). The core of the process is an interaction between these two kinds of modelling with a continuous dialectic inter-flow of information. Classifications and descriptions are an integrated part of the data modelling process and as such they are active research components in their own right. It is inherent to this process that we vary and change our data models to be able to confront our theoretical models with new and different data structures. The moment we subside and accept a particular classification as everlasting the research process stops. We thus need classifications and descriptions to change continuously. Further we need alternative competing and supplementary classifications to be available for any particular set of data.

As stated previously, a working version of IDEA has been established and tested. The testing has not been extensive, but it has been sufficient to give us some idea of the advantages and disadvantages of IDEA.

If we take the advantages first, then indeed we have succeeded in creating a system that gives us a high degree of flexibility in recordings. It has been easy for us to customise it to reflect a particular recording practice and to create exactly those qualifiers needed to describe a particular material. In this sense the system works exactly as intended.

Another aspect, which has proven very efficient and easy to use, is the recording of relationships internally and externally between instances of the five basic units. This recording takes place as a simple selection of the instances to link, and what is normally a very cumbersome work in any traditional excavation system, seldom carried through to any depth, is almost ridiculous simple in IDEA. At the same time it is worth remembering that we are dealing with true many to many links between all entities.

A third aspect, which has proven very efficient, is the retrieval of data. This does of course speak more of Microsoft Access than of IDEA, but it is worth noting that despite the strongly fragmented table structure of IDEA it is possible to create very well structured reports as well as all sorts of tabular query extracts with very little work.

The disadvantages may, however, be of more interest and certainly of more consequence than the advantages – and there are disadvantages.

One disadvantage that we have taken lightly is the reaction of would-be users, when they start on IDEA and realise that there is nothing there, so to speak. They expect to be able to start right away doing recordings, but find no way to go ahead and fill in data. The notion that they themselves can design the system as they wish may appeal to them, but they tend to back out, when they realise that they have to learn to do so first. This problem can be overcome of course. We can deliver predefined versions that subsequently may be adjusted, and IDEA has been provided with export and import facilities to help move complete sets of definitions from one database to another.

Another disadvantage seems to be of a much more serious nature. In a traditional database all fields for data entry are predefined, and each field in a form is directly associated with a field in a table, mostly so that fields in the same form is associated with fields in the same table. Data entry can thus be made a fairly straightforward matter, where you just bend your head and type away with little or no need to use the mouse.

This is certainly not possible in IDEA. As a consequence of its design, most data recording in IDEA consist of linking existing records in different tables to each other. Consider the following example: You have to enter find x, and record that it is of type y and was found in context z. In a very simple recording system you would simply have a form with three fields. In the first you would type x in the second you would type y and in the third you would type z. In IDEA it is different. Initially in the finds form you will have to type in x, then in a subform often using several clicks with the mouse you have to

select y among the types of an already defined classification scheme. Finally, with a click of the mouse you bring up a form where, with another couple of clicks with the mouse, you set a link to context z.

I am very aware of the drawbacks of this. Not only is the extended use of the mouse not healthy at all, but it is also infinitely slower than using the keyboard. During the design of IDEA we did not pay very much attention to this. We do now. With the latest version of Microsoft Access, which includes some ingenious tree-view controls, we can reduce the number of clicks needed considerably, but we cannot remove them altogether. The way IDEA is designed with its many linked tables do secure a high degree of data integrity and a very high degree of flexibility in the hands of the user. It prevents, however, the existence of all-inclusive data entry forms where you just bend your head and type away.

One way out of this problem, of course, would be to design data entry front-ends for specific configurations of IDEA. This, however, would limit the flexibility of recording, and it would certainly go against the basic philosophy of the system, whereby all customisation should be end-user based without any needs for physical changes to the system.

This kind of problem with the user interface is not specific to IDEA. As soon as we wish to utilise the full potentials of relational database management systems, as we have done in IDEA, these types of problems will occur. We cannot entirely avoid them, I believe, but in living with them we have to find ways to limit their influence.

The user interface problem certainly was not a reason to stop further developments of IDEA along the lines of its original design. We did, however, stop for an entirely different reason. The table design had a build in flaw that would eventually block further development. To understand this we have to return to our basic design.

With five entities fully inter-linked, our basic design consisted of 14 tables. It was fairly simple, and nothing to be worried about. For each qualifier we added to the entities, however, we not only had to add the one or more tables constituting the qualifier, we also had to add a minimum of five tables to link the qualifier to the entities using many to many links. The number of tables grew rapidly, and by the end of our development of version 1 the system contained 85 heavily inter-linked tables.

Version 1 of IDEA only covered what can be termed the textual part of excavation documentation. We did have an entity for drawings, but it was only meant to handle the textual information associated with the drawings, not the drawings themselves. Our intentions were, however, to expand IDEA to cover vector based information of the drawings as well. This should be kept in tables of the database in such a way that there would be a fully operational linkage between the tables holding the vector based drawing elements and the basic entities they described.

We had been through it all, we knew what it would take to do it, but we simply had not counted on how many extra tables we would need to add this feature to our existing

design. And beyond that there could be other features we would like to add, which would mean even more tables. In addition to this, our code, which had already grown considerable and become more and more cumbersome to maintain, would virtually explode. Right there and then we decided to re-evaluate our complete design.

The result of our re-evaluation was a totally changed design. Unfortunately, we did not have any more time within the limits of the project to implement this new design. Over the last year, however, I have been experimenting a little with it to see if it could be made to work.

To understand what our bearings are now, you have to ask yourself: What makes different entities different, if all their qualifiers are stripped from them and placed in separate tables? The answer is simply nothing. An entity is an entity, is an entity. After all, the tables of our five basic entities only consisted of an ID field for different instances of the entity. Further you may consider what is the difference between an entity and a class? An entity may somehow be considered to be just a super-class. What separated them in our original design was that entities have instances, but classes have none. They were just qualifiers to the instances of the entities.

So perhaps what we needed was simply a table for entities and classes and a table for instances. To this would come a table for relations, something that was also very important to version 1 of IDEA. Further I have found it practical to add a table for variables, but this could probably have been incorporated with the entities/class table as well.

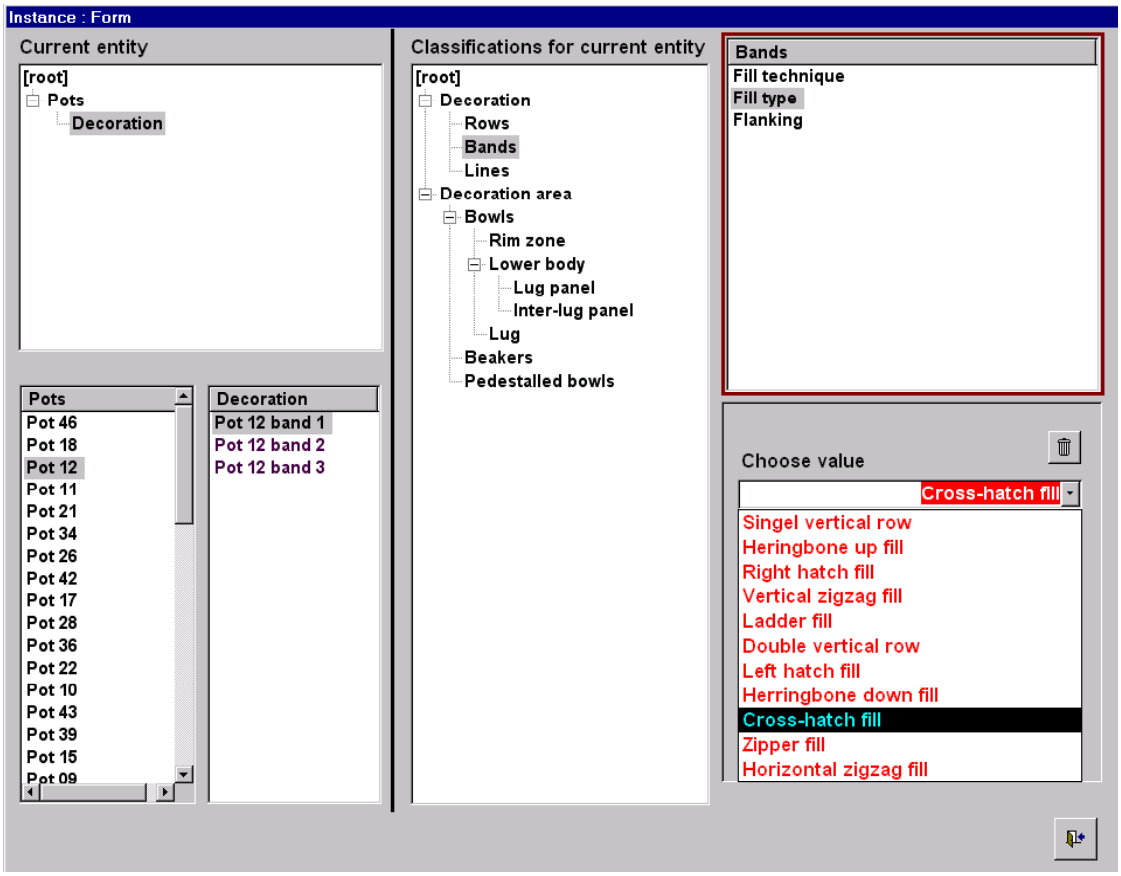
To this basic design then comes one or in a few cases two tables for each data type available for recording. It should be noted that each table or in some cases pairs of tables can be considered a data type. Thus composite types like event, point, line, area, etc. may be created with little difficulty. Importantly, these types may be added without any serious consequences to the overall number of tables.

I have not tried to implement an excavation recording system in this structure. Instead, since it is now a completely generalised recording system, I have implemented part of a description system for pottery, which by itself could have been part of a excavation system.

It is worth noting that apart from the basic definition of entities, classes and variables and apart from the linking of instances with each other, all data entry in this system may be carried out through one single form. Such an entry form may appear as shown below. Its main features can be summarized as follows:

- Any number of recording entities may be created, and organized into one or more parallel hierarchies. The structure of these is taken into account by the system. In this case the entities are decorations and pots.
- For each instance of higher-level entities (pots in this case) any number of instances of subordinate entities (decoration in this case) may be recorded.

- For each entity any number of hierarchically organized classification systems may be applied, and each instance of the entity may be classified using different classification systems simultaneously.
- For each class of the classification systems any number of variables may be defined.



To conclude my paper I should like to make the following points:

IDEA is not *the* solution to a generalised recording system for excavations. It is a step in that direction, and to my knowledge the first serious step where generalised recording is meant to be understood as freedom of choice to record as you wish.

New versions, taking the IDEA experiences into account, should be attempted. As I have just tried to show, it is in fact possible to create very generalised table designs and make them work including making efficient searches of the resultant structures. The problem is not whether we need these kinds of systems, but whether we can find acceptable ways to interface them and work with them on a daily basis.

In my opinion we must take this direction in our design and do it now. I know that with the current versions of available database management systems like Microsoft Access we do not have the perfect tool for this endeavour, but right now it will do. There will be new versions of Access or other database management systems headed in an object-oriented

direction. There is one overriding reason why we have to take up this challenge. We have to develop our recording systems in a way, where they can support our efforts to do the best possible research.

Cited literature

Andresen, J. & Madsen, T. 1992: Data Structures for Excavation Recording. A Case of complex Information Management. C.U. Larsen (ed*.): Sites & Monuments. National Archaeological Records*. The National Museum of Denmark, pp. 49-67

Andresen, J. & Madsen, T. 1996a: IDEA – the Integrated Database for Excavation Analysis. H. Kamermans and K. Fennema (eds.): *Interfacing the Past. Computer Applications and Quantitative Methods in Archaeology CAA95*. Analecta Praehistorica Leidensia 28, Leiden, pp. 3-14.

Andresen, J. & Madsen, T. 1996b: Dynamic classification and description in the IDEA. III International Symposium on Computing and Archaeology. *Archeologia e Calcolatori* **7**, pp. 591-602.

Madsen, T. 1988: Determining priorities in Archaeology – an introduction to a debate. *Arkæologiske udgravninger i Danmark* 1987, pp. 24-27

Madsen, T. 1995: Archaeology between facts and fiction: the need for an explicit methodology. M.Kuna & N. Venelová (eds.): *Wither Archaeology? Papers in Honour of Evzen Neustupný*, Praha, pp. 13-23.