Archeologia e Calcolatori
10, 1999, 125-144

COPING WITH COMPLEXITY. TOWARDS A FORMALISED METHODOLOGY OF CONTEXTUAL ARCHAEOLOGY

1. Introduction

A paper dealing with the implementation of description systems for archaeological data probably does not appear very relevant to most archaeologists. After all, implementing a description system is only a matter of drawing up a table with the items to be described (the units) in the rows and the descriptive elements (the variables) in the columns. This can be done on a piece of paper, in a spreadsheet, or in a simple database. There may seem to be no more to it than that, but are archaeological data always of a nature that makes simple tables the best choice? Few archaeologists have considered this question, but most, I believe, have experienced that it is not always easy to make data fit into a flat table. Indeed, how do you describe complex composition of decoration elements on pots, the spatial interrelationships of artefacts in a grave or the intricate combination patterns of flint refits, to mention just a few straightforward examples?

Contextual archaeology has become a concept in recent years. We have to understand and interpret the archaeological evidence in and by its contexts. We just cannot extract the evidence from the record and interpret it on its own, or more to the point in our own right. Carefully we have to study the context of the evidence and draw our interpretations from the set of interrelationships that we disclose (Hodder 1986, 118-146; 1992, 14-15; 1997). A plea for a contextual archaeology is certainly very just, but it is not something you decide and then do overnight. There is one major obstacle to overcome, and that is how to do a proper description of contextual information. How can you interpret something if you cannot describe it properly? Naturally, I am not referring to our ability to describe in prose a particular set of relationships – that of course is trivial. The difficulty lies with formalised and systematic description cutting across a series of contexts disclosing in union the perhaps most important type of contextual information of all, that of coherent contextual structure.

Archaeology has come next to nowhere in describing complexity, and as long as archaeologists rely on the same old ways of describing data in flat tables, it will move nowhere. There are other and more powerful ways to describe data, however, ways.that will allow a better handling of complexity. Modern relational databases, if exploited to their full potential, can cope with types of data that cannot be described properly in simple tables. The aim of this paper is to investigate the general structure of archaeological data from a recording point of view. I will look at basic issues like the interrelationships of entity types, types/classes and variables in descriptive systems, and I will look at the basic characteristics of context information. Further, I shall give an example of a generalised recording system that will handle both simple data and the more difficult contextual data.

2. Basic concepts in data recording

2.1 *Entity types and relationship types*

2.1.1 Entities

The concept of entities is not often used in archaeology. In probably the best book on archaeological typology written, W.Y. and E.W. Adams use it «to designate whatever is classified and/or sorted in a typology» (Adams, Adams 1991, 340). Entities thus refer to the items – whether physical or conceptual – being classified, which upon classification become type members (Adams, Adams 1991, 32). Their definition is in full accordance with the way the concept is used in relational database theory. «An entity... is a "thing" in the real world with an independent existence. An entity may be an object with a physical existence – a particular person, car, house, or employee – or it may be an object with a conceptual existence – a company, a job, or a university course» (Elmasri, Navathe 1989, 40).

In order to cope with entities in relational databases, it is customary to split them into groups of entities that share the same set of attributes (variables). Such groups are called entity types (Elmasri, Navathe 1989, 42). An entity type usually has an attribute that must have a unique value for each individual entity of the type. This is called the key attribute and may be seen quite simply as a numbering device for the entities of a particular type.

2.1.2 Entity types

In archaeology we normally do not use the concept of entity type. Strictly following database theory it is easy, however, to separate what should be considered entity types in a recording system. It would be the combination of what we number uniquely and what we attach a unique set of variables to. If we were doing research on artefacts from burials on a number of cemeteries we would be likely to attach unique numbers to the cemeteries, to the graves and to the artefacts in the graves. Each of these would then be an entity type. If further we created descriptive systems for the various artefact categories – pots, swords, brooches, beads, etc. – these would also become entity types because they would not share the same variables. For each of these entity types a separate table would then be needed in a relational database system.

Since it is hardly likely that two researchers doing research on cemeteries would use the same descriptive systems for the artefacts, each investigation would demand its own unique relational database for recording purposes. This is one of the reasons why the use of databases in archaeological research is very much *ad hoc*, and why any attempt to introduce database standards in archaeology leads to intolerable inflexibility (MADSEN 1998, 1999).

For reasons that may not appear immediately clear at this point I prefer to restrict the entity type concept in archaeology to cover groups of entities that are united by their numbering device, the need for which must be decided by archaeological arguments. Let it suffice to say that the demand for separate entity types in association with different sets of variables, and hence for a unique numbering of these sets of entities has to do with specific rules of standard relational database implementation. Using an object oriented database implementation there is no need to separate these as entity types. In the above example there would thus only be the cemeteries, the graves and the artefacts as entity types. If, however, we wished to carry out a more detailed study of the decoration composition of pots in the graves we would by all probability need to apply relationship types. In that case the decoration elements also need to become an entity type with a unique numbering, for as we shall see in the following, relationship types only apply to entity types.

2.1.3 Relationship types and relationship instances

A relationship instance is a specific association between two or more entities. A relationship type is a set of relationship instances that share the same value of association (ELMASRI, NAVATHE 1989, 46). For all practical purposes in archaeology we may limit the relationship type concept to cover associations between two entities.

Entities associated by relationship types can be from both different entity types and from the same entity type. In an excavation recording system entities from the entity types "photos" and "contexts" may be associated by the relationship types "shows" and "is shown on". Likewise in an excavation recording system entities from the entity type "contexts" may be associated with each other by the relationship types "lies above" and "lies below". It may be noted that in both examples we have two relationship types applying to an identical pair of relationship instances. Which one to use depends on the direction from which the relationship instance is viewed. Such asymmetric relationship instances are the most common, but symmetric ones also exist. As an example we may take the entity type "finds" from an excavation recording. Some of the entities here may be pottery shards, which on conservation may be glued together. Thus we have the relationship type "fit together" to describe a relationship instance between two shards glued together. It will be the same from no matter which shard we view it.

Relationship types can also have attributes similar to those of entity types. We may for instance have the relationship type "found in" to associate the entity type "finds" with the entity type "contexts". Due to the nature of excavations there are often uncertainties with this relationship type. It may not always be possible to state for certain that a particular find stems from a particular context. To cope with this we could add an attribute to the relationship type where the certainty of the association could be stated.

Relationship type is a formal concept of relational database theory for which there are straightforward ways of implementation. In archaeology the concept of relationship type is equivalent to the concept of contextual information at large. The use and importance of contextual information has always been a central issue in archaeology. Scandinavian archaeology saw a major dispute by the end of the preceding century between Sophus Müller (MÜLLER 1884) and Oscar Montelius (MONTELIUS 1884) for and against the overruling importance of contextual information. The nature of contextual information discussed then, like things "found together", was fairly simple and easy to record. There are other types of contextual information that are much harder to record even if we may recognise their existence and importance.

2.1.4 Relationship type constraints

Relationship types may or may not have certain constraints that limit the possible combinations of entities participating in relationship instances. There are two possible constraints. Cardinality denotes the number of relationship instances that an entity may participate in, and participation specifies whether the existence of an entity is dependent on being related to another entity via the relationship type or not. Thus a relationship instance may either be mandatory or optional. Together we refer to these as structural constraints, and they may manifest themselves in a number of ways (ELMASRI, NAVATHE 1989, 50-51).

For all practical purposes we need only consider three structural constraints in archaeological recording[1]. One of these is that no constraints exist between two entity types. They may be recorded independent of each other and all entities of both entity types may participate in an unlimited number of relationship instances with entities of the other entity type (the cardinality is many to many). If we take the entity types "photos" and "contexts" in an excavation recording then obviously any photo may show many contexts (or none at all) and any context may be shown on many photos (or on none at all).

[1] I do not take a one to one cardinality into consideration here. The reason for this is that this cardinality is normally used where different variables apply to different sets of entities, that otherwise would be from the same entity type. As already argued, an object oriented approach to database implementation removes the need for a separation of entity types on this basis, and hence also the need for the use of the one to one cardinality.

The two other structural constraints share a cardinality of one to many but is divided by whether a relationship instance is mandatory or optional. If optional, relationship instances need not be established between the entities of the two entity types. If they are established, however, the entities of the one entity type can only participate in one relationship instance, while entities of the other entity type may participate in an unlimited number of relationship instances. If for instance you wished to do a study of a particular artefact type then probably you would make a recording of the type members in an entity type we could call "objects". You would also make a recording of the find circumstances of the object – their "contexts". Now you would have a situation where you would always record the objects, but as you may have objects without any contextual information, an object need not participate in a relationship instance with "contexts". If however you do have contextual information for an object, then naturally the object may only participate in one relationship instance with "contexts". Any context on the other hand may hold several instances of your artefact type, and consequently an instance of "contexts" may participate in several relationship instances with "objects".

In the third structural constraint the relationship type is mandatory for an entity of one of the entity types to exist. In this case there will be a cardinality of one to many with the one side on the independent entity type and the many side on the dependent entity type. There are numerous examples of this type of structural constraint in archaeology. Actually, it is the most common structuring element used in recording systems. An example could be the previously mentioned example where we would be doing research on artefacts from burials on a number of cemeteries. For each cemetery, there would be a number of graves, and for each grave there would be a number of artefacts. Each artefact would have to belong to a grave and each grave would have to belong to a cemetery. It should be noted that used extensively this type of constraint would lead to a hierarchical breakdown of a material to be described.

2.2 *Typologies and variables*

2.2.1 Typologies

The literature on arrangements[2] of archaeological data is vast, complicated and often self-contradictory. For the purpose of discussing the struc-

[2] Arrangement is here a direct translation of the Nordic word *ordning*, which in Scandinavian archaeology of the nineteenth century was used to denote any kind of procedure to put data in some sort of order. I prefer this fairly neutral word as a general reference to all the more loaded terms we use like categorization, classification, series, taxonomy, typology, etc.

ture of archaeological data recording it is not necessary, however, to enter the dark sides of this literature. At the stage of recording we apply arrangements, we do not create them.

Any arrangement used in a description system has the purpose of sorting entities of a particular kind into mutually exclusive categories. Adams and Adams (1991, 47) reserve the word typology for this kind of arrangement, and I shall do the same here. There are three basic requirements for a typology. It must have clearly defined boundaries, it must be comprehensive within its boundaries, and it must have a sorting mechanism that leads to mutually exclusive typehood (Adams, Adams 1991, 76-77).

The boundaries for any typology used in a description system are ultimately determined by the entity type to which it is attached. By definition the typology cannot handle anything outside the entity type. On the other hand it is fully possible for a typology to handle only part of an entity type. This happens when the boundaries of a typology are defined by a previous typology. One type may here define the boundaries for the next typology. Think of the entity type "finds" from an excavation recording. You may have a typology that splits this entity type into "small finds", "bag finds" and "samples". Subsequently you may have a typology that divides the category "small finds" into various categories of artefact types. One of these may be "axes", which in yet another typology may be split into different types of axes. What you have is a series of typologies that provides a hierarchical breakdown of a material. For each level in the hierarchy the boundary of the typology is narrowed down based on the typology of the previous level.

As for the comprehensiveness and sorting mechanisms this is obviously an inherent part of any typology. If these demands are not met it is simply not a typology. This does not mean that we cannot find "typologies" used in recording systems that are neither comprehensive nor possess adequate sorting mechanisms. In practice "anything goes" when you set up a recording system, but if your typologies are not properly defined, you will sooner or later run into problems with inconsistencies in your recordings.

2.2.2 Variables

A variable may be viewed as a feature or characteristic, which varies from one entity to another, and which is taken into account in the definition and/or description of types (Adams, Adams 1991, 370). In database terminology a variable is called an attribute (Elmasri, Navathe 1989, 40-42). This is confusing to archaeologists as the term attribute normally is used in archaeology to designate discrete values within a variable. Variables may be used to describe entity types directly, or to describe types within typologies used to qualify entity types.

The variable concept is also used in such fields as statistics and computer programming from where we may adopt some useful formalisation. A variable is characterised by having a name, a domain, a type and a value set. The domain is equal to the set of entities to which the variable may be applied, whether being the total of an entity type or a subset based on a typology.

The variable type is treated very much along the same lines in statistics and computer science, but with different results. In statistics it is customary to distinguish between nominal, ordinal, interval and ratio scale types (see for instance SHENNAN 1988, 10-13). This is a division based on the kinds of arithmetic operations that may be applied to the data (Nominal scale: =; Ordinal scale: = > <; Interval scale: = > < + -; Ratio scale: = > < + - * /). In computer science the type division is based on what can be considered valid operations on the individual types, and differences in semantics of the same operations on different types. There are a lot more types distinguished in computer science than in statistics, but this is due to the wider scope of computer science compared to statistics, not to a fundamentally different approach to type division.

In a computer based data recording system it is natural that we have to comply with the types of computer science such as boolean, byte, integer, real, char, string, etc., but it does not mean that we cannot create types for recording adapted to our recording needs. Again if we focus on an object orientated approach to database recording we may create types customised to our needs by combining some of the basic types available for recording together with some rules for how these combinations should be understood and be operated upon. We could, for instance, create types for points, lines and areas in space, or we could create types for persons or for archaeological dates.

The value set of a variable is simply the total range of values that a variable may assume. The nature of this value set will vary considerably with the type in question from the simple true and false of the boolean type to the continuos and in principle unlimited number of values of the real type.

2.2.3 Typologies and entity types

You may have noticed that there is something peculiar about the relationship between typologies and entity types. Both are parts of a breakdown – often hierarchical – of the data to be described, and both may have descriptive variables directly associated with them. The basic difference is that entity types have entities in their own right – instances that are numbered and accounted for – typologies don't. Typologies are merely qualifiers to entity types much the same way that variables are. Indeed typologies are little more than logical "pigeonholes" based on specific values of a set of variables. Small as this difference may seem it nevertheless plays a decisive role. Relationship types can only exist between entity types because only entity types contain "things" – the entities – that may be explicitly referred to. So whenever we need to record contextual information we have to make sure that we are using entity types, not typologies. The choice of something being an entity type or a typology must be made during the design of the recording system. It is a choice that does have consequences for what can and what cannot be recorded. Later I shall present an example where decoration bands on pots are viewed as an entity type, because I need a relationship type between the individual bands to handle relational information. I might just as easily have defined the bands as part of a typology, which is the normal approach, but then the relational information between the bands would elude me.

3. Experiments with generalised recording systems

3.1 *IDEA - Integrated Database for Excavation Analysis*

The inspiration for a generalised excavation recording system using the full potentials of relational databases came up in the late eighties during discussions between myself and Jens Andresen, based mainly on a profound knowledge of relational databases that Jens had already acquired. We found such a system possible if a set of universal entity types could be agreed upon, and if in principle any entity could be freely linked internally and externally to all other entities of these entity types (Andresen, Madsen 1992).

From 1994 to 1997 in a three-year project we were given the possibility to attempt an implementation of our ideas (Andresen, Madsen 1996a). We found that we could indeed implement our basic design in such a way that the system could be customised to meet the varied recording requirements of different archaeologists without interference with the table design. Further, we found that we could create additional tables to the system organised in such a way that users could define their own variables attached to the various entity types, meeting the specific needs of a particular excavation. We even found a way to implement a universal, dynamic system for classification and description with user defined types and variables (Andresen, Madsen 1996b). The result of the project was a flexible and reasonable bug-free recording system for archaeological excavations with good search and report facilities. We called this system IDEA.

The IDEA project provided us with many valuable experiences, both of a positive and a negative nature. One very important thing we learned was, that although it is standard in relational database theory to require a separate table for each set of entities, to which a unique set of variables is attached, there are other ways to do things. Especially our experiment with user-defined typologies created on the fly with associated descriptive variables was a breakthrough. It became obvious that using an object orientated approach to database design we could create recording systems that were powerful and flexible with very little ado.

On the negative side we found that the five basic entity types we had decided upon heavily restricted our design. If users needed other entity types than those that we had defined they would not find the system useful. Worse, however, was that the design turned out to be very complicated to maintain and almost impossible to expand. In version 1 there are more than 80 heavily interrelated tables. We had originally planned a version 2 that could cope with vectorised drawings as part of the database recordings in addition to the textual information of version 1. To do this, however, would have added so many extra tables to the database that it would have become unmanageable. We were actually heading straight into a *cul de sac* as far as further development of the system was concerned.

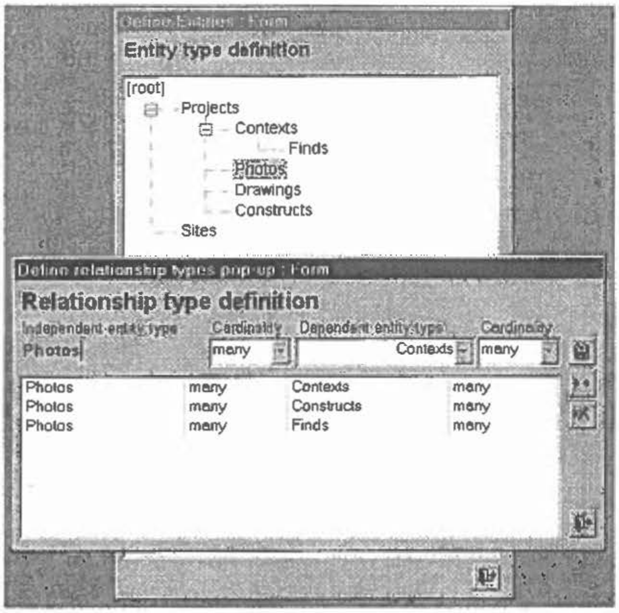
3.2 *GARD - General Archaeological Recording Database*

The last few month of the IDEA project we used to discuss and outline the design principles for a new and much different system. There was no time left within the IDEA project to implement such a system, but on and off over the last couple of years I have been experimenting with an actual design and implementation of a generalised recording system. As part of this work I have also tried to clarify (not least for my own good) some theoretical aspects of archaeological data recording as outlined in the first part of this paper.

The system, which I have tentatively called GARD, is very different from IDEA in its structure. Although I shall not discuss its design here it should be said that only some twenty tables are involved, a quarter of the number in IDEA. Further, the amount of code written is a mere tenth of what was used in IDEA. All the same GARD is a much more powerful and versatile system than IDEA, first of all because it has no predefined entity types.

GARD allows you to define any number of entity types. You do this in a form where you can place the entity types into parallel or hierarchical structures (Fig. 1, background). As you do, relationship types are automatically created with a one to many cardinality down hierarchies and with restrictions that will only allow you to create an instance of a lower ranking entity type if it is bound to one of the "parent" entity type. For those entity types that are not bound in this way you may freely define relationship types. Double clicking an entity type will bring up a form, where you can set all relationship types logically possible in relation to the entity type you activated (Fig. 1, foreground).

Having defined the entity type structure, various other definitions may be entered. Thus you may define the naming of relationship instances (above – below, shows – seen on, etc.); variables to be used for the description of



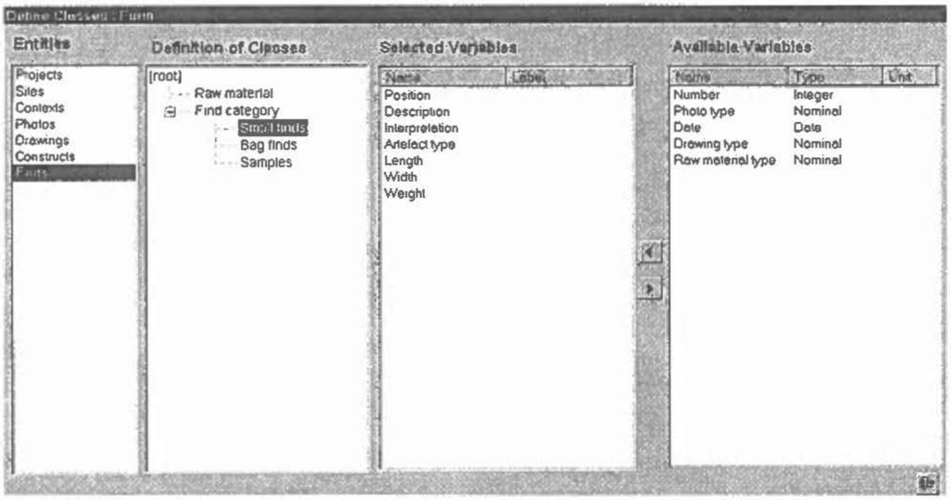
Independent entity type	Cardinality	Dependent entity type	Cardinality
Photos	many	Contexts	many

Photos	many	Contexts	many
Photos	many	Constructs	many
Photos	many	Finds	many

Fig. 1 – Screen dump from GARD. Background: form to create and modify the entity type structure for a recording. Foreground: pop-up form to define relationship types for the entity types.

entity types, whether directly or through the use of typologies; and the typologies to be attached to entity types.

The latter definition takes place in a form with four list- and tree-view controls (Fig. 2). In the first from the left the entities defined are listed. You may here select an entity, on which all subsequent operations in the other controls are carried out. In the second control you may define class structures for the chosen entity. You can add and delete classes, and you can move individual or complete branches of classes. For each class you can add variables by moving them from the list of available (= previously defined) variables to the list of selected variables, or you can remove already selected variables. Thus you can personalise each class with a selection of variables describing it. You can also add variables directly to the "root" of the classification tree, and thus as descriptors to the entity type itself. Adding a class anywhere in a classification tree will automatically lead to an inheritance down the tree, so that all classes in branches below the actual class will be attributed the same variable. Likewise, if you delete a variable it will be deleted in classes down the tree. There are however important constraints to these operations. You cannot remove a class or variable from an entity type or restructure a class hierarchy if it has already been used to record actual data.


Define Classes : Form

Entities
- Projects
- Sites
- Contexts
- Photos
- Drawings
- Constructs
- Finds

Definition of Classes
```
[root]
  - Raw material
  ⊟ Find category
      - Small finds
      - Bag finds
      - Samples
```

Selected Variables

Name	Label
Position	
Description	
Interpretation	
Artefact type	
Length	
Width	
Weight	

Available Variables

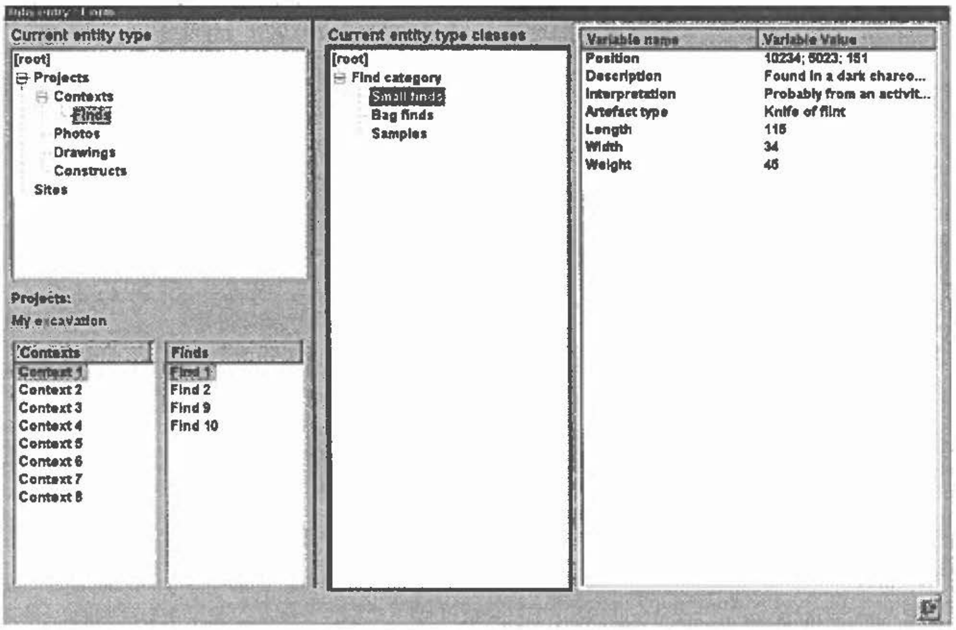
Name	Type	Unit
Number	Integer	
Photo type	Nominal	
Date	Date	
Drawing type	Nominal	
Raw material type	Nominal	

Fig. 2 – Screen dump from GARD. Form to define classification schemes for the entity types, and to attach variables to entity types and to individual types of the classifications.

Data entry takes place through the form shown in Fig. 3. It contains five tree- and list-view controls. In the upper left-hand corner you may select the entity type. Below you will find entities of the current entity type to the right and entities of its parent entity type to the left. Since all entities in a hierarchical structure has to be tied to an entity of the parent entity type (unless it is the root) you cannot go directly to the entity type and select an entity as the system simply will not know which entities to display. If you look at the screen-shot you will see that in the space between the controls it is noted that the listings relates to the project "my excavation". Thus to get to the situation of the screen-shot a sequence of selections of entities down the entity type hierarchy has been carried out.

Selecting an entity makes the associated classification system appear in the central control of the form. Selecting a class will make all its variables with whatever values recorded appear in the rightmost control. To edit the values of the variables you double click the class name and get a pop-up form, in which you may enter, change and delete values (Fig. 4). The fields in this form have the same order of appearance as in the parent form, an order that you may edit in the class definition table.

To set relationship instances between entities you double click an entity in the data entry form from which you want to set a link. This will bring up a form that looks a little like the main entry form (Fig. 5). In the upper left corner the entity from which the form was activated is written. This is the entity linked form. The form lets you select the entity to link to as well as



Current entity type

[root]
- Projects
  - Contexts
    - Finds
  - Photos
  - Drawings
  - Constructs
- Sites

Projects:
My excavation

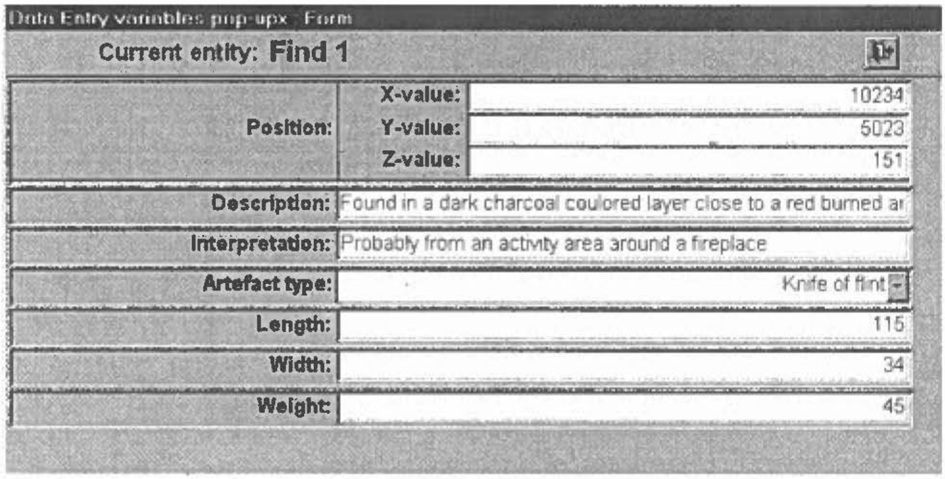
Contexts	Finds
Context 1	Find 1
Context 2	Find 2
Context 3	Find 9
Context 4	Find 10
Context 5	
Context 6	
Context 7	
Context 8	

Current entity type classes

[root]
- Find category
  - Small finds
  - Bag finds
  - Samples

Variable name	Variable Value
Position	10234; 5023; 151
Description	Found in a dark charco...
Interpretation	Probably from an activit...
Artefact type	Knife of flint
Length	115
Width	34
Weight	45

Fig. 3 – Screen dump from GARD. Entry form for recording entities as defined by the entity type structure. Selecting a class of the classification bound to the current entity type will display the attached variables and their values.



Data Entry variables pop-upx : Form

Current entity: Find 1

Position:	X-value:	10234
	Y-value:	5023
	Z-value:	151
Description:		Found in a dark charcoal coulored layer close to a red burned ar
Interpretation:		Probably from an activity area around a fireplace
Artefact type:		Knife of flint
Length:		115
Width:		34
Weight:		45

Fig. 4 – Screen dump from GARD. Form to enter and edit variable values for an entity type or for a particular class attached to an entity type. The form is activated by double clicking the class in question (or the root of the classification).

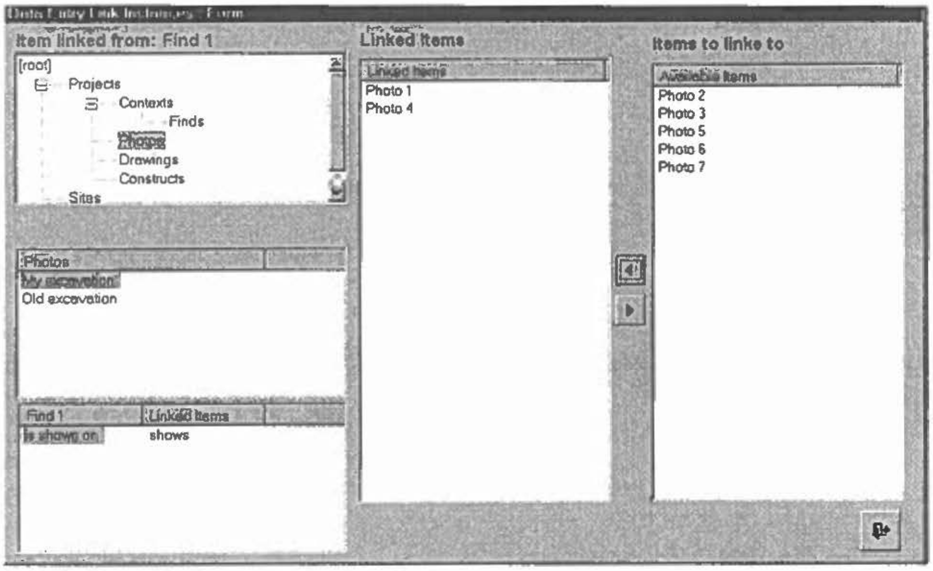


Fig. 5 – Screen dump from GARD. Form to set relationship instances between entities of different or the same entity type. The form is brought up from the main entry form (Fig. 3) by double clicking an entity that has to become part of a relationship instance. The system controls that only "legal" relationship instances can be defined.

the name of the relationship instance. The entity to link to must be located first. This is done in much the same way as in the main entry form. In a tree-list control you can navigate down through the entity type tree, selecting on the way the entities that is the parent of the next selection. As you progress, the entities of the current entity type (selected in the entity type tree) will be displayed in the two list boxes on the right. The one on the left hold those entities that have already been linked to the base item of the form and the other hold those that are still available for linking. The constellation of entities in these two controls will also depend on the relationship type selected in the control in the lower left corner. There may be one or more relationship types named here, and for each of these a link to an item may be set. If there are no relationship types visible (none has been defined) there will be no entities displayed in the two rightmost controls, and no linking can take place. To set a link or delete a link you simply select an entity in one of the two controls and move it to the other using the buttons with arrows between the controls.

4. Relational description - contextual archaeology: a challenge for the future

At the beginning of this paper I commented upon contextual archaeology and the need to find a way to handle complex relational information formally. Thus a notion of contextual archaeology implies that you are capable of describing, recording and analysing relationships. It may not seem a big deal, but when archaeologists are confronted with many to many cardinality between relationship types, they tend to stop short of comprehension when the "how to" turns up.

The issue is simply not easily understood, and there are only a few areas within archaeology where attempts to cope with the complexity of relationships are seen, like stratigraphic relationships in deeply stratified sites, or refitting of flint for instance. Even if analytical methods are quite well developed in these cases, the initial basic recording of the relationships is not. Mostly it boils down to adding a list of items linked to the item in focus together with some information on the nature of the relationship that is the relationship type. To carry out subsequent analyses you either have to rely on manual partly intuitive procedures, or if you use computerised methods you have to rewrite the recordings into a format useable for the programs. Mostly this is a listing of relationship instances, where each instance is described by its first entity, its relationship type and its second entity. The GARD system is meant to be one possible solution that archaeologists may choose, if they want to be able to implement a proper description of relational information.

As mentioned, stratigraphic relationships and refitting of flint are two very obvious areas of using relational description. It was areas that immediately sprang to eye, when we first discussed the potentials of relational databases in archaeology (ANDRESEN, MADSEN 1992). There are numerous other areas of use, however. Two early papers by Costis Dallas point to two of these: the description of interrelationships of objects and/or features in space, and the description of composition in decorations on artefacts. He demonstrates how relational description, as he terms it, may be used to describe complex house plans and the composition of human figures on Classical Attic grave reliefs (DALLAS 1992a, 1992b).

In the following I shall provide another example of relational description applied to decoration composition. I have deliberately chosen a very simple example; or rather I have simplified something which in its totality is actually rather complex. I have done so in order to demonstrate that even simple composition cannot be described efficiently unless you use relational description.

My example draws on the beautifully decorated bowls from the early Middle Neolithic TRB-culture in South Scandinavia (Fig. 6). A dominant feature of their decoration is the vertical bands below the rim and neck decoration. Some 30-40 bands are normally present around the pot divided into two sets by lug panels. The composition of these bands is clearly structured, and stylistic studies of the bowls normally focus on this structure. It is

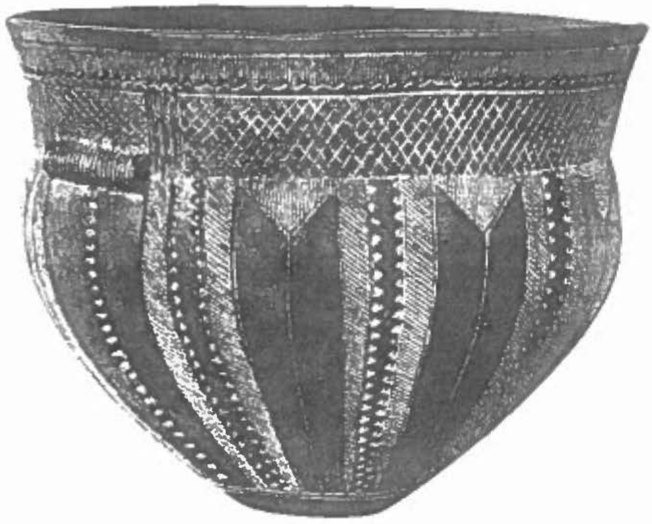


Fig. 6 – Band decorated bowl from the Danish early Middle Neolithic (after MÜLLER 1918).

customary to view the composition as consisting of triplets of bands, where "empty" bands separate the triplets. Traditional description consist of a list of possible band triplets, where you may note the most common (often there is just one or a few on each pot). Whenever there is more than one triplet combination, or indeed when triplets are not applicable, the descriptions become inaccurate. Further, by way of definition "empty" bands are "separators", and thus completely removed from description. Thus you cannot investigate the actual role of empty bands in the composition.

The only way to cope with this type of band composition is to take each band in turn, describe it and note what other bands lies to its left and right respectively. You may believe you can do this by extending the traditional way of flat table recordings and reserve one column for each band in their order of appearance. However, if you start thinking about it, you will quickly realize the futility of this venture. Even if you should accept the number of columns it would require to describe a complete bowl, it would all be of no use. You simply cannot compare different pots based on this sort of recording. There are several problems, but one stands out immediately. In a fragmented material, how would you decide, which bands present on the fragments should go with which columns of the recording sheet?

The solution, and indeed the only solution, is to use relational description. If we take three bands in succession you may acknowledge that the following sentence can be seen as a full description of the bands in Fig. 7: (**herringbone band** *made with* **flint edge impressions** *flanked by* **incised**

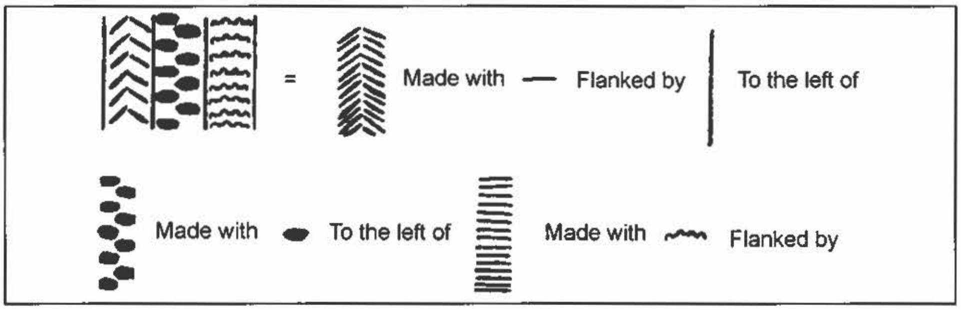


Fig. 7 – Formal decomposition of a triplet band composition.

lines) *to the left of* **(zipper band** *made with* **heavy oval stabs)** *to the left of* **(ladder band** *made with* **cardium impressions** *flanked by* **incised lines).**

There are no less than three different relational descriptors in this sentence. If we wished to take all these into account we would have to separate three entity types: band fill pattern, band fill technique and flanking. This would be unpractical. Instead we will use only one entity type – band – and view band fill pattern, band fill technique and flanking as variables of this entity type. The decomposition would then take the following form: **band** (fill: herringbone pattern; technique: flint edge impression; flanking: incised line) *to the left of* **band** (fill: zipper pattern; technique: heavy oval stabs; flanking: none) *to the left of* **band** (fill: ladder pattern; technique: cardium impressions; flanking: incised lines).

Making the band an entity type means that each band on each pot has to be identified by a unique number. You may not need this number for any other purposes than the creation of the relational description. Thus if you do not need to be able to identify the individual bands later on in your analyses in the same way as you need to be able to identify the individual pots, you can simply forget about it, once the data entry is done.

The result of the recording will be that for each pot a number of bands (depending on how much of the pot is preserved) will be described by three variables. Each band will be linked to one or two other bands as being either to the left or right of these. For each pot you will thus get one or more "chains" of bands. You can traverse these chains in both directions starting at any band you choose.

In a certain sense relational description like this can be considered to be dynamic. We cannot use the information as it is, because there is no way to present it in one single format suitable for analysis. On the contrary there are many possible ways of representation. We have to make a structured extraction of data, and depending on how we do this we may get very different types of data with very different potentials. It is not the purpose of this

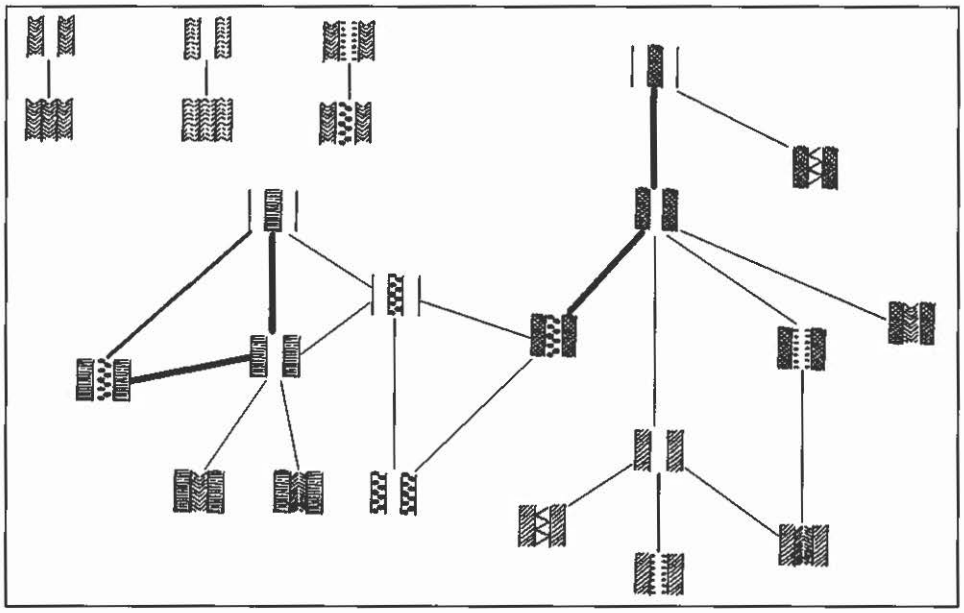


Fig. 8 – Graph showing the appearance of symmetric triplet band combinations on 32 Neolithic bowls. For each pot where a combination of two different triplet combinations is seen a count is made. The thickness of the lines between triplet combinations indicates on how many pots the combination is seen.

paper to discuss neither the different ways to extract data, nor the different ways to analyse these data. I should however point to one obvious way of analysing and displaying relational information of this type, namely through the use of graphs. This method is already well known in connection with stratigraphical analyses and refitting analyses. In connection with the band decoration of the Neolithic bowls, it may be demonstrated with an example from a trial recording of 32 pots (Fig. 8).

No description system, and certainly not one involving relational description, can be said to lead to an objective recording. With each and every system come decisions of what exactly to record and how to record it. When dealing with relational information there is a further major choice to be made of the syntax of description. This may not be very obvious with stratigraphic information nor with data on refitting of flint because we feel quite sure that we know the "true syntax". When we turn to something like decoration composition we are in a completely different field. Our decision of how to dissolve the composition into structural elements, and our decision of how the basic syntax of the composition should be viewed can create completely different recording systems. Some good examples of very different ways of viewing syntax in Neolithic pottery may be seen in HODDER 1982 and VAN DE VELDE 1980. However, in all cases the problems of describing relational information is the same and the solution to the description problem the same, the use of relational description systems.

5. CONCLUSION

Contextual information has always been a central issue in archaeological research. In the preceding century it was intensely discussed, and influential researchers claimed that it was the most important type of information in archaeology (MÜLLER 1884). In this century context information has been commonly accepted as one of the cornerstones in archaeological methodology. A major problem has been, however, that given a piece of paper and a pencil there is a limit to the complexity of contextual information that may be recorded in a systematic manner. As a result the use of contextual information in archaeology has never gone beyond a certain level of complexity. Many have tried, I believe, to get to grips with the more complex aspects of contextual information, but failed. The plea for a contextual archaeology – a renewed focus on contextual information – by influential archaeologist of the last couple of decades (eg. HODDER 1986, 1992, 1997) has led practically nowhere. At most we have seen studies using complex contextual information on a particularistic basis. Far-reaching conclusions are drawn based on selected information from a few sources – contextual information used out of context so to speak.

If we wish to progress from this situation, we have to find ways to handle context information in its full complexity and not just as fragments detached from the broader framework. This implies two things. Firstly, we have to find ways to describe contextual information that will retain its structural complexity in all aspects. Secondly, we have to develop analytical methods that can cope with the complexity of contextual information, or rather relational data as they have turned into through the description phase.

Hardly anything has been done in archaeology to solve these problems. They can be solved, however, and it is no excuse that we cannot make it work with pencil and paper. The computer is by now available to all archaeologists, and both description and analysis can be developed to a high degree of perfection using this media. The first step is to find suitable ways to carry out the descriptions. The tools in terms of modern database technology have been available for years now, so it is time we started using them.

The present paper draws attention to the problem of describing contextual information using an object oriented approach to relational database techniques. Initially, it outlines the basic theoretical concepts for a structured description of complex information. The insight gained from this exercise is used to demonstrate how a generalised solution may be implemented. The solution, the GARD system, which has been developed based on experiences gained through the IDEA project, is not unique in any way. There will be other ways to reach a similar solution, if only researchers will start focusing on the problem.

The next step will be the analytical methods. Nothing much has been done here either. In his papers on relational description DALLAS (1992a, 1992b) suggested the use of similarity coefficients to compare entities based on a multitude of partly contradictory relationship instances. Another solution is to use some kind of graph related methods, as done in Harris matrices, and as demonstrated in this paper with the Neolithic band decorated bowls. In these cases the relationship types are simple and unambiguous, and the formatting of the graph easy, but even in cases where the logic is not simple, where ambiguous, contradictory relationship instances occurs, it should be possible to apply graph-related methods. There is a lot of research going on in various disciplines referred to as network analysis, trajectory mapping, ordinal networks, semantic networks, etc. (KRUSKAL, WISH 1978; KLAUER 1989; RICHARDS, KOENDERINK 1995) that it will be worthwhile for archaeology to study in some detail.

TORSTEN MADSEN
Department of Archaeology
University of Aarhus - Moesgård

Acknowledgements

I should like to thank Jens Andresen, Mads Holst and Nick Ryan for comments on an earlier version of this paper. Especially, I am grateful to Nick Ryan for using his knowledge of computing and relational databases to save me from a few blunders.

BIBLIOGRAPHY

ADAMS W.Y., ADAMS E.W. 1991, *Archaeological Typology and Practical Reality. A Dialectic Approach to Artifact Classification and Sorting*, New York.

ANDRESEN J., MADSEN T. 1992, *Data structures for excavation recording. A case of complex information management*, in C.U. LARSEN (ed.), *Sites & Monuments. National Archaeological Records*, Copenhagen, 49-67.

ANDRESEN J., MADSEN T. 1996a, *IDEA - The Integrated Database for Excavation Analysis*, in H. KAMERMANS, K. FENNEMA (eds.), *Interfacing the Past. Computer Applications and Quantitative Methods in Archaeology - CAA95*, Analecta Praehistorica Leidensia, n. 28, Leiden, 3-14.

ANDRESEN J., MADSEN T. 1996b, *Dynamic classification and description in the IDEA*, in P. MOSCATI (ed.), *III International Symposium on Computing and Archaeology*, «Archeologia e Calcolatori», 7, 591-602.

DALLAS C. 1992a, *Relational description, similarity and classification of complex archaeological entities*, in G. LOCK, J. MOFFETT (eds.), *Computer Applications and Quantitative Methods in Archaeology - CAA91*, BAR International Series, n. 577, Oxford, 167-178.

DALLAS C. 1992b, *Syntax and semantics of figurative art: a formal approach*, in P. REILLY, S. RAHTZ (eds.), *Archaeology and the Information Age: A Global Perspective*, London, Routledge, 230-275.

ELMASRI R., NAVATHE S.B. 1989, *Fundamentals of Database Systems*, Redwood City.

HODDER I. 1982, *Sequences of structural change in the Dutch Neolithic*, in I. HODDER (ed.), *Symbolic and Structural Archaeology*, Cambridge, Cambridge University Press, 162-177.

HODDER I. 1986, *Reading the Past. Current Approaches to Interpretation in Archaeology*, Cambridge, Cambridge University Press.

HODDER I. 1992, *Theory and Practice in Archaeology*, London, Routledge.

HODDER I. 1997, *'Always momentary, fluid and flexible': towards a reflexive excavation methodology*, «Antiquity», 71, 691-700.

KLAUER K.C. 1989, *Ordinal network representation: Representing proximities by graphs*, «Psycometrika», 54, 4, 737-750.

KRUSKAL J.B., WISH M. 1978, *Multidimensional Scaling*, Quantitative Applications in the Social Sciences, n. 11, Beverly Hills.

MADSEN T. 1998, *Digitaliseret registrering af udgravninger - er der brug for datastandarder og fælles strategier?*, in H.J. HANSEN, V. ØDEGAARD (eds.), *De nordiske museer og informationsteknologien - rapport fra en konference 1.-3. December 1996*, TemaNord, 513, København, 167-177.

MADSEN T. 1999, *Digital recording of excavations - do we need data standards and common strategies?*, in H.J. HANSEN, G. QUINE (eds.), *Our Fragile World. Recording the Past for the Future. National Museum*, Copenhagen.

MONTELIUS O. 1884, *Den förhistoriska fornforskarens metod och material*, «Antikvarisk Tidskrift för Sverige», 8, 3, 1-28.

MÜLLER S. 1884, *Mindre Bidrag til den forhistoriske Archaeologis methode*, «Aarbøger for Nordisk Oldkyndighed og Historie».

MÜLLER S. 1918, *Stenalderens Kunst i Danmark*, København.

RICHARDS W., KOENDERINK J.J. 1995, *Trajectory mapping: A new non-metric scaling technique*, «Perception», 24, 1315-1331.

SHENNAN S. 1988, *Quantifying Archaeology*, Edinburgh, Edinburgh University Press.

VAN DE VELDE P. 1980, *Elsloo and Hienheim: Bandkeramik Social Structure*, Analecta Praehistorica Leidensia, n. 12, Leiden.

ABSTRACT

The present paper draws attention to the problem of describing contextual information using an object-oriented approach to relational database techniques. Initially, it outlines the basic theoretical concepts for a structured description of complex information in a relational database. The insight gained from this exercise is used to demonstrate how a generalised object-oriented solution may be implemented using a standard relational DBMS. The implementation called GARD is an all-purpose recording system, where the user can create a particular database structure through its interface without changes to the underlying table structure, and modify the database as needed parallel to the recording of data. Finally, an example using decorated bowls from the Danish Neolithic shows how complex relational information may be handled. This information has been entered into GARD and extracted again for analysis.