A TARGET LOGICAL SCHEMA: THE ACS

P.M. STOCKER* and R. CANTIÉ**

*UNIVERSITY OF EAST ANGLIA, NORWICH NR4 7TJ, UNITED KINGDOM
**MATRA, TOULOUSE, FRANCE

Abstract

This extended abstract describes a data structure (ACS) intended to serve as a target for conceptual schema languages and as a source for implementations. It indicates some of the uses to which it has been put.

1. Introduction

Over the past decade two related trends have become apparent. The first, in the database area, is to capture some of the semantic meaning of the data, and to store this also in the database by means of a 'conceptual schema'. The second, in the area of programming languages, is the move to non-procedural programming, complex data types whose structure is not revealed to the user, constraints embedded in data types and so on. Both demonstrate the transition to the situation where internal structure and representation within the computer are no longer the concern of the programmer, who works increasingly at a logical level.

A conceptual schema language has two roles, one is that of providing a language which allows its user to make the transition from the model of the universe of discourse to defined, representable data items. This role serves to define the relationships between the data items of concern and the constraints upon them, and also provides a supportive structure which helps the user to resolve his concept of the transition. The second role is to capture and store these relationships and constraints in a form which relates to the purpose of providing a basis for an implementation of the database. This abstract is concerned with the second role, that is, with the mechanism for retaining the information captured by the conceptual schema language rather than the language itself.

In the transition, through the various layers, from universe of discourse to stored bytes of basic data, explicit information disappears. It is either stored within a layer, to be reincorporated on the outward path, or it becomes implicit in a data structure[1]. This abstract proposes a small set of structural forms which suffice to contain the information obtained through the conceptual schema language, the set is called the Abstracted Conceptual Schema (ACS). It is intended to serve as a target for conceptual schema languages.

2. The Abstracted Conceptual Schema

The ACS is based upon two logical structures, the classical set and an indirect functional mechanism called the element set.

All data values are defined in terms of dataitem types; a dataitem type is a unitary item of data which is not further divisible for any purpose; it is a logical item, e.g. 'salary in dollars'; information concerning representation (internal and external) is additional to the ACS. Information concerning comparability, that 'salary in dollars' may logically be compared with 'tax in dollars', is part of the ACS.

The element set provides the functional association between dataitems. An element set type is a set of function/dataitem type combinations. For brevity the function/dataitem type combination will be called a property and, when a value of the dataitem is associated with it, we shall refer to a property value. A graphic representation is shown in Figure 1, f_i, d_i and p_i refer to functions, dataitems and properties respectively.

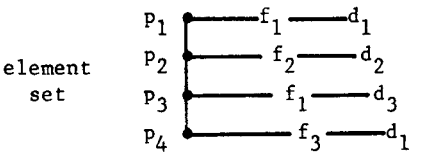

Figure 1

Elements of the same element set type are contained in an element set. Thus an element occurrence is a tuple of property values (e.g. d_1, d_2, d_3, d_1 in Figure 1). Properties are divided into identifying properties (identifiers) and non-identifying properties. An identifier uniquely determines an element within an element set, i.e. only one tuple within the set takes this value.

The element set mechanism, therefore, has the following result. There is a functional relationship between the values of an identifying property and the values of each other property of the element set type. This mechanism is equivalent to the Nijssen 'bridge', expressed in n-ary rather than binary terms. Identifying properties relate to LOTS, non-identifying properties to NOLOTS.

The element set is one way of capturing the relationships established by the conceptual schema language. Relationships are also implied where two element sets have a common property, or, (weaker) a common dataitem type.

3. Additional Structures

3.1 A combination of properties may be used, called a compound property, and a corresponding identifier defined.

3.2 The transmission of properties through a hierarchy (or network) of element sets is incorporated (using matched identifying properties). This provides the structure necessary to capture 'roles'.

3.3 The constraint mechanism is implemented by means of classic set theory. The set of values taken by any property (or combination of properties) may be formed into a set of data-item values (or tuple values), called a value set. A value set may be defined on one element set and then used to constrain the allowable domain of property values taken in another element set, by means of set identity or containment operators. The normal operations of set theory may be used to construct new value sets, from values of comparable dataitem type.

4. Representation of the ACS

The ACS is a number of data structures, not a language. It could be represented in many ways. The representation which has been used so far is in terms of relations. It is representable (for site to site transmission, for instance) in terms of 15 relations; of these 9 are trivial in the sense that they correspond to the name declarations of a programming language, dataitem type names, function names, etc. The relations which carry structural content are:

i) definition of properties of element sets
ii) definition of identifying properties of element sets
iii) definition of value sets
iv) application of value sets as constraints
v) construction of value sets by operators
vi) specification of 'role' networks.

5. Utility of the ACS

The presentation of the conceptual schema as data rather than procedure appears to have at least tactical merit.

5.1 It has been assessed as a target for a number of conceptual schema languages discussed at the CRIS-1 conference[2], NIAM, CIAM and ACM/PCM; an automatic both-way mapping with ADAPLEX[3] is operational, as is a mapping from CODASYL DDL. No difficulties were met in these mappings and the set based constraint mechanism was found to be capable of capturing more procedurally expressed constraints than had been anticipated.

5.2 The structural form avoids arguments which arise in the supportive role of conceptual schema languages concerned with distinctions between entities and relationships, since both map into similar element structures. Moreover, the 'special role' of time is transmuted into conventional functional relationships.

5.3 It forms a basis for a comparison between schemata obtained from conceptual schema languages and schemata describing implementation, e.g. an ADAPLEX schema with the CODASYL DDL for its implementation. In this context it has been adopted as the basis for a heterogeneous distributed database project, PROTEUS.

5.4 It has proved convenient as a basis for a number of varied current research projects, e.g. unification of schemata which overlap, 'intelligent' aids to form construction.

5.5 It may serve as a point of discussion on possible open network architecture protocols, since it allows information interchange in a relatively uncoloured and non-committed form[4].

5.6 The relational form of representation is highly convenient as one of the inputs required for automatic implementation.

6. Information Representation

It would, in many instances, be advantageous if a model of given information content transformed into the same ACS structure, irrespective of the conceptual schema language used in the process. The ACS itself has semantic rules. Additional rules may be added to control the splitting of element sets by definition as against the provision of a property which achieves the same effect. So it is possible that the ACS might be constrained into unitary, orthogonal constructs. These comments apply to structure; the manner in which the conceptual schema languages make use of names prohibits equivalent naming.

References

1. P.M. Stocker. "Canonical Schemata, Canonical Queries and Standardisation", Infotech State-of the Art Report, Series 9, Number 8, Database.

2. Information Systems Design Methodologies: A Comparative Review. Proc. IFIP TC8 Working Conference on Comparative Review of Information Systems Design Methodologies, Noordwijerhout, The Netherlands, May 1982.

3. M.J. Smith, P. Bernstein, U. Dayal, N. Goodman, T. Landers, K.W.T. Lin and E. Wong. "Multibase-integrating Heterogeneous Distributed Database Systems", AFIPS, Chicago, 1981.

4. Concepts and Terminology for the Conceptual Schema and the Information Base. J.J. van Griethuysen. ISO TC97/SC5/WG3 - N695, 1982.