A UNIFIED ALGEBRAIC VIEW
OF STRUCTURED SYSTEMS DEVELOPMENT MODELS

T.H. Tse *
London School of Economics
University of London

1. INTRODUCTION

Numerous models have been proposed under the name of structured systems development. Examples are data flow diagrams [7, 10], Jackson structure diagrams, Jackson structure text [14], system specification diagrams, system implementation diagrams [15], Warnier/Orr diagrams [17], and Yourdon structure charts [22]. They are widely accepted by practising systems developers through the simplicity of use and the ease of communication with users. But because of the lack of a common theoretical framework, transition from one model to another is arbitrary and can only be done manually. Users tend to stick to a particular model not because of its superiority but because of familiarity. Automatic development aids tend to be ad hoc and model-dependent.

To solve the problem, there is a need to provide a formal/theoretical link for the structured models. In the proposed dissertation, an initial algebra approach [3, 4, 13] is used. An algebra will be defined and linked to DeMarco data flow diagrams, Jackson structure text, and Yourdon structure charts. These three have been chosen because they represent three distinct forms of structured systems development models.

2. ADVANTAGES OF A UNIFIED ALGEBRAIC VIEW

A unified algebraic view of the structured models is useful for several reasons:

(a) Specifications can be transformed from one form to another through signature morphisms, equations, and derived operation symbols.

(b) Different structured models are suitable for different situations depending on the environment [19], emphasis [5], and stage of development [16]. But it has been found that individual models may not be used in some installations because the users are not familiar with them [1]. With a transformation system, the most suitable model can be used, independently of user familiarity.

(c) In recent years the initial algebra approach has been used extensively in the specification of abstract data types. Examples are Clear [2, 18] and OBJ [9, 11]. Although they are not originally intended for structured models, the interpreters for abstract data types can be adapted for validation of our specifications and the transformation between different models.

(d) Automatic development aids for one structured methodology can be applied to another method through transformations. The development aids described in [6], [8], and [20], for example, may be extended to other structured models.

* Current address: Department of Computer Science, The University of Hong Kong, Pokfulam, Hong Kong.
3. ALGEBRAS

Intuitively, an algebra is a family of objects that satisfy a formal structure. To define an algebra \( A \), we must first define the formal structure through the concept of a signature. A signature consists of a set \( S \) of object types, known as sorts, together with a family \( \Sigma \) of sets, each set containing operation symbols that connect the sorts. We shall use \( \Sigma^{s_1 \ldots s_n, s} \) to denote the set of operation symbols that connect the sorts \( s_1, \ldots, s_n \) to the sort \( s \).

Given the skeleton structure, we then complete the definition by relating it to real objects. Each sort \( s \) is mapped to a set \( A^s \), which is called the carrier of \( s \). Each symbol \( q \) in \( \Sigma^{s_1 \ldots s_n, s} \) is mapped to a function

\[ q_A : A^{s_1} \times \ldots \times A^{s_n} \rightarrow A^s \]

which is called an operation.

Let us apply the algebraic fundamentals to structured systems. Conceptually, as structured system is specified by a hierarchy of tasks. Each task consists of a name and a structure, together with the interfaces. The structure indicates whether the task is elementary, or is made up of subtasks in the form of sequence, selection, iteration or parallelism. The interfaces are in the form of data flows, which are input/output data related with other tasks, files, and the environment.

The signature for structured systems, then, consists of a set \( S \) of six sorts: task, name, struct, inter, flow, data, and a family \( \Sigma \) of sets of operation symbols:

\[
\Sigma^{\text{name inter struct}^n, \text{task}} = \{ \text{task} \} \\
\Sigma^{\text{task}^n, \text{struct}} = \{ \text{seq, sel, par} \} \\
\Sigma^{\text{task}, \text{struct}} = \{ \text{itr} \} \\
\Sigma^{\Lambda, \text{struct}} = \{ \text{elem} \} \\
\Sigma^{\text{flow}^n, \text{inter}} = \{ \text{io} \} \\
\Sigma^{\text{data}, \text{flow}} = \\
\{ \text{in, out, inflag, outflag, infile, outfile, source, sink} \}
\]

for any positive integer \( n \) and where \( \Lambda \) is the empty string.

The sorts of the signature are mapped, respectively, to the set of tasks, the set of task names, the set of structures, the set of interfaces, the set of data flows, and the set of data names. The symbols are mapped to operations explained as follows:

(a) The operation \( \text{task}_A \) specifies the name, structure and interfaces of a task.

(b) The operations \( \text{seq}_A, \text{sel}_A, \text{itr}_A, \) and \( \text{par}_A \) link up a number of subtasks into a structure.

(c) The operation \( \text{elem}_A \) indicates that a structure is elementary, i.e. it does not consist of subtasks.

(d) The operation \( \text{io}_A \) indicates that the interfaces are made up of a number of data flows.

(e) The input operations \( \text{in}_A, \text{inflag}_A, \text{inflag}_A, \) and \( \text{source}_A \) are used to name the data passing through a data flow into a task. They also indicate, respectively, that the data are inputs from some other task, flags from some other task, inputs from a file, and inputs from the
Different algebras can be defined over the same signature. Homomorphisms, or functions preserving the signature, can be defined from one algebra to another. Such homomorphisms enable us to forget about minor syntactical differences and concentrate on the major issues in a specification. For example, an expression like "seq(a; b; c)" in a particular algebra may be mapped to an alternative algebra giving

```
a seq
b; c
a end
```

which is in the notation of Jackson structure text. Furthermore, if we allow operation symbols to vary through signature morphisms and/or derivation of symbols, the expressions can be mapped to more graphic forms giving DeMarco data flow diagrams or Yourdon structure charts.

4. INITIAL ALGEBRA

The algebra that has the richest context is called an initial algebra, denoted by $A_0$. It has the property that, given any other algebra $A$ over the same signature, there exists a unique homomorphism mapping $A_0$ to $A$. Because of this guarantee of homomorphisms, we would like to use the initial algebra approach to link up various structured systems development models. A initial algebra for structured models is defined thus:

4.1 Carriers

We regard task names and data names as more fundamental than other variables in our algebra, because these names appear unaltered in the final specification. We shall *enlarge* the signature by putting in task names and data names as *elementary symbols*. Let $X$ denote the enlarged set consisting of all operation symbols, elementary symbols, as well as three delimiter symbols: “(”, “;”, and “)”. The carriers of $A_0$ are made up of *terms* in $X$, i.e. strings of symbols from $X$. We define the carriers $A_0 <s>$ by induction as follows:

(a) For any elementary symbol $q$, if it is a task name, we let the term “$q$” be in $A_0 <name>$, otherwise we let the term “$q$” be in $A_0 <data>$.

(b) For any operation symbol $q$ in $\Sigma<s_1 \ldots s_n>$, and for any terms $u_1$ in $A_0 <s_1>$, ..., $u_n$ in $A_0 <s_n>$, we let the term “$q(u_1; \ldots; u_n)$” be in $A_0 <s>$.

4.2 Operations

Operations $q_{A_0}$ in $A_0$ are induced from the symbols $q$ as follows:

(a) For any elementary symbol $q$, we define $q_{A_0}$ to be the term “$q$”.

(b) For any operation symbol $q$ in $\Sigma<s_1 \ldots s_n>$, and for any terms “$u_1$” in $A_0 <s_1>$, ..., “$u_n$” in $A_0 <s_n>$, we define $q_{A_0}(u_1, \ldots, u_n)$ to be the term “$q(u_1; \ldots; u_n)$”.

3
It can be shown that the algebra $A_0$ thus defined is an initial algebra. It can be mapped by homomorphisms to other algebras over the same signature. Furthermore, if we allow signatures to vary, more user-friendly forms will result.

5. PROPOSED RESEARCH

It is proposed that structured models such as Yourdon structure charts, DeMarco data flow diagrams, and Jackson structure text be linked up using homomorphisms, signature morphisms, equations, and derived operation symbols. Specifications can thus be transformed from one form to another. A prototype system is further proposed to test the feasibilities. As a result of the research, the most suitable model may be chosen for a target system, independently of user familiarity. Algebraic interpreters may be used to validate the specifications. Automatic development aids for one methodology may be applied to another.

REFERENCES