Journal of Universal Computer Science, vol. 7, no. 8 (2001), 668-691
submitted: 18/5/01, accepted: 21/8/01, appeared: 28/8/01 © *Springer Pub. Co.*

An Eclectic View of the Irish School of Constructive Mathematics $M^{\clubsuit}_{C}$, from [Lucas 1978] to [Mac an Airchinnigh 2001]

Mícheál Mac an Airchinnigh
(University of Dublin, Trinity College,
mmaa@cs.tcd.ie)

Abstract: In this paper I celebrate the evolution of the Vienna Development Method (VDM) along its Irish branch and attempt to tell the story that Peter Lucas played in it.

There are two parts to the paper. In the first part I tell my story of the early day of the origins of the Irish School of the VDM (VDM$^{\clubsuit}$), beginning with pre-history in 1978 up until the radical decisions of 1995 which led to the Irish School of Constructive Mathematics ($M^{\clubsuit}_{C}$).

In 1995 the School committed itself to the development of the modelling of (computing) systems in full generality. This was achieved by embracing Category Theory and by exploring a geometry of formal methods using techniques of fiber bundles. From fiber bundles to sheaves was a natural step. Concurrently, the School moved from the algebra of monoids to categories, and from categories to topoi (*alt.* toposes). The second part of the paper illustrates, with simple examples, how I introduce topos logic into modelling in 2001.

Key Words: constructive mathematics, intuitionistic logic, modelling, topos theory, VDM.
Category: G, F.4

1 Prologue

> "*We stress here, as was stressed in the introduction to this volume, that the meta-language is to be used, not for solving algorithmic problems (on a computer), but for specifying, in an implementation-independent way, the architecture (or models) of software. Instead of using informal English mixed with technical jargon, we offer you a very-high-level 'programming' language. We do not offer an interpreter or compiler for this meta-language.* ***And we have absolutely no intention of ever wasting out time trying to mechanize this meta-language. We wish, as we have done in the past, and as we intend to continue doing in the future, to further develop the notation and to express notions in ways for which no mechanical interpreter system can ever be provided.*** [emphasis is mine]" [Bjørner and Jones 1978] p. 33.

These words had a profound effect on me and on the development of the Irish School of the Vienna Development Method (VDM$^{\clubsuit}$). I remember being an angry young(er) man and throwing them back verbatim in the faces of their authors, Dines Bjørner and Cliff Jones, in the 'basement' of the Berlaymont headquarters of what was then known as the

European Economic Community (EEC) in the heart of Brussels, Belgium. The occasion, as I remember it, concerned the launching, or at least the promotion of the International Standards Organization (ISO) standard VDM-SL (where SL abbreviates specification language). Both Dines and Cliff were very much in support. I was adamantly and vociferously against it. I still am! [*As I write this now in 2001 and use words such as 'remember' I know that my account is strictly a story (= h*istory*). I may get some facts terribly wrong. However, the spirit of what I remember is close enough. Perhaps others will correct the inaccuracies. Certainly it is better to have this story than none at all.*]

The lengthy quotation heading the prologue is taken from the first substantial account of the Vienna Development Method published in 1978 in the (new) Springer-Verlag *Lecture Notes in Computer Science* **61**. I bought the book in that same year and first read words by someone called Peter Lucas. Permit me to quote the last three sentences from his introductory paper, *On the Formalization of Programming Languages: Early History and Main Approaches*.

> "*So far most research in formal semantics has been concerned with constructs as found in traditional languages. Here is a piece of language, what does it mean, was the question in the light of the discussed software problems. We should start from the other end, i.e. construct novel denotations and associate a name after we are satisfied with their properties.* " [Lucas 1978] p. 23.

To put this another way, Peter Lucas was basically requiring of us that we carry out a programme of research into constructive mathematics. Ultimately, this is what has led to the VDM$^{\clubsuit}$ and subsequently and necessarily to the Irish School of Constructive Mathematics ($\mathrm{M}_{\mathrm{C}}^{\clubsuit}$).

Who *is* this man Peter Lucas? Why am I writing this contribution to the *Festschrift* in his honour? Naturally, it is an opportunity to explain one's own (intellectual) insights of the time (which I date as 18–19 May 2001, the occasion of the Colloquium to mark his retirement from the Institute for Software Technology, Graz University of Technology, Austria) and to try to relate same in the context of the one to be honoured (Peter Lucas). But the insights and this writing (*Schrift*) must be correlated with one's experience of the man whose (intellectual) life we celebrate. Elsewhere within the collected writings of the *Festschrift* there shall be a variety of accounts and perspectives of Peter Lucas, the man and his work. The perspective of each story teller (narrator, writer, ...) will have space-time threads each of which is a sequence of events

$$EVENT^{\star},$$

dated appropriately

$$EVENT \longrightarrow DATE,$$

and pegged to location

$$LOCATION \longrightarrow (EVENT \longrightarrow DATE),$$

either real or virtual, and the perspective will be peculiar to the story teller (her/him)-self. Each story teller will relate their understanding of Peter Lucas within their own conceptual framework and according to the roles and relationships that have characterised their own life and their relationship directly or indirectly with the one to be honoured.

In this *Schrift* I wish to honour him from the perspective of his influence on my own intellectual development and of all that work which now goes by the name of the *Irish School of Constructive Mathematics* ($M_C^{\clubsuit}$), incorporating the Irish School of the VDM. I hope that it will be clear from what I write that Lucas and I have had a very special relationship throughout the past 24 years! That relationship was based upon a very few chance meetings which took place in strange countries (exclusively in Europe). Over those 24 years we have had a few conversations and touched upon quite a range of topics. Some few of those conversations were related to the VDM. In fact, so rarely did I meet him that I shall often speak of those who knew him and worked for him, especially Dines Bjørner and Cliff Jones. I may have occasion to speak of others such as Andrzej Blikle whom I remember with great affection and who is certainly one of those who worked in the research direction, advocated in [Lucas 1978], from denotations to syntax [Blikle 1987] [Blikle 1990]. Such people come within a collection of persons that I would associate with all that happened in and that came out of the IBM Vienna Laboratory of the nineteen-seventies. I certainly met, albeit briefly, Erich Neuhold and learned of Hans Bekic († October 1982) and Wolfgang Henhapl.

Perhaps the most significant collection of persons with whom I might now identify when I think of Peter Lucas is that once named VDM Europe (established in 1985?). For the very first VDM Europe Symposium (1987) there were four members of the Programme Committee (and four editors): Dines Bjørner, Cliff Jones, Erich Neuhold and the stranger/outsider: Mícheál Mac an Airchinnigh. I was the substitute for the invited Hans-Jürgen Kugler. The rest is now history. Subsequently VDM Europe collectively agreed to evolve into Formal Methods Europe (FME). At the FME Symposium in 1994 (Barcelona, Spain) I lobbied that Peter Lucas might become our chairman and so it became and became us and became him. I subsequently met him again at FME 2001 in Berlin, Germany, in 2001.

I most certainly read his writings before I met the man. I had thought that our first meeting was at the VDM Europe Conference in Kiel, Germany in 1990. I was wrong! Peter told me that we met in Brussels in 1987. Most significantly was his presence at all of the three tutorials that I have presented at the Symposia: 1987 in Brussels, Belgium, 1991 in Noordwijkerhout, the Netherlands and 2001 in Berlin, Germany. (I wonder will he make it to the next one?) Thus was Peter Lucas well-placed to experience at first hand the emergence and the evolution of the $\text{VDM}^{\clubsuit} \subseteq M_C^{\clubsuit}$ over 14 years. But first let me tell my story, starting with a cycling trip in Germany that brought me to Köln and to a bookshop on the 14th of August 1978.

" *. . . in fact our only way to talk scientifically about the relation of humans to*

their natural languages is in terms of computer notions (or so it seems to me)."
[Lucas 1978] p. 3.

I had just completed my primary degree (BSc) in pure mathematics with the University of London and was about to embark upon a master's degree (MSc) in computer science with the University of Dublin, Trinity College (1978–79). It was in this shop in Köln that I bought my first book on the VDM. I wasn't searching for it. I was merely curious to see what sort of mathematics texts German university students might be using in the Köln region. I was indulging in my browsing habit. The VDM text was newly published, written in English, and was entitled *The Vienna Development Method: The Meta-Language*. My guess is that I purchased the book precisely because it was volume **61** in the now very successful series *Lecture Notes in Computer Science* (LNCS). I had never come across the LNCS before that.

I can see myself now, tall, lean, very long mid-shoulder length hair (cut in January 1979), full beard and no spectacles, in cycling shorts dusty from the roads of Germany standing there in the shop, taking the book down from the shelf, flicking through it quickly, being impressed with the apparently easy mathematical notation scattered throughout. I bought it there and then, put it into the paniers of my racing bike and set off for the *Conference in Mathematics Education* which was about to take place in Ghent, Belgium. (I suppose the shop owner in Köln was used to selling mathematics books to Irish 'hippies' on racing bicycles!) I arrived exactly on time at the youth hostel in Ghent where I received two items waiting for me—a telegram to say that I had been successful in my London University Degree examinations and a very large package in a box containing my suit, shirt, tie, shoes, socks, etc. for the Conference and the Banquet. The Youth Hostel staff were very surprised at my appearance the following morning. I was very surprised at how difficult the LNCS **61** book on the VDM really was.

It was a full 5 years later when I began to see the light, when I began to understand, and all thanks to one man who gave a lecture in a very crowded room that seated 20 in Pearse Street, Dublin. (There were about 50 of us, with barely standing space, at the lecture. Most were the 4th year undergraduate students in our Computer Science degree programme.) The guest speaker was Dines Bjørner and the date was the 30th of November 1983. (I think the date is right! Dines signed my copy of LNCS **61** and dated it thus: DUBLIN 30 XI 83). I was a junior lecturer in Computer Science at the University of Dublin, Trinity College, and had obtained most of my serious understanding of the subject whilst at Graduate School in the State University of New York (SUNY) at Stony Brook (1979–80). I was already quite familiar with algebraic specifications thanks to my German colleague Hans-Jürgen Kugler. Dines presented a VDM specification of something or other as a collection of the usual domain equations. Then he covered up the right-hand side (where the VDM *modelling* details were given) and, of the left-hand side, said that this was just like algebraic specifications. The difference being that the VDM presented specific models of everything. That did it. My blindness was cured.

2 Enter the VDM♣ on Stage in Noordwijkerhout in 1991

> "*When I create models, I am the Creator, I am God.*" [Peter Gorm Larsen's memory (Reinhardt's Restaurant, near Unter den Linden, Berlin, 11th March 2001) of the opening words of the author at the first Irish VDM Tutorial in the deconsecrated circular chapel of the Noordwijkerhout conference centre in 1991. This was indeed the first one. The naming of the Irish VDM only took place in 1990.]

Looking back over ten years to that time of the first public presentation of the Irish School [Mac an Airchinnigh 1991] it is interesting to ask what was it exactly that constituted the school? In what manner was it different from traditional VDM?

Then I declared that "a formal (development) method is essentially constructive applied mathematics", and therefore that "the starting point of all work must be the problem domain" and that this modelling ought be be considered analogous to the formulation of partial differential equations (**pde**s). In particular it was emphasised that just as there were both closed form solutions and approximations to **pde**s then one ought to expect similar results in formal modelling.

The School had method as well as mathematics. More importantly it had a specific philosophy which was sketched out in *Conceptual Models and Computing* [Mac an Airchinnigh 1990]. Most significantly, Irish VDM was deemed to have an inner world or self-contained reality of its own.

> "*the application of the Irish School of the VDM is opening up a whole new branch of constructive mathematics. In other words, there is a perspective of the School that has* ***overtly*** *nothing whatsoever to do with the specification, design and implementation of systems.*" [Mac an Airchinnigh 1991].

Among those in the very small audience taking the tutorial were Peter Gorm Larsen, Dines Bjørner and Peter Lucas. The chapel looked very empty. In a parallel tutorial session Jim Woodcock had the much larger audience. Such is life! Dines was one of my Doctoral Thesis examiners (Stephen Goldsack of Imperial College, London, was the other.) and so might be expected to have a vested interest in being there. Both Peter Gorm Larsen and Peter Lucas were keenly interested in VDM, or at least so it seemed to me then.

2.1 Method

> "*In applied mathematics, the use of the standard models expressed as partial differential equations was undoubtedly a major factor in scientific progress*" [Mac an Airchinnigh 1991].

In 1991 it seemed that everything could be built out of five standard models. (Products of the form $A \times B$ were then considered to be too obvious to need mention.) Moreover, a model consisted on some structures and some operations on those structures. The basic structure was the monoid.

For any set X one could apply the powerset operator to obtain the set of all the subsets of X, denoted $\mathcal{P}X$. From a practical constructive point of view it was always assumed that X would be finite. I certainly did not agonize over $\mathcal{P}\mathbb{N}$. There were two isomorphic structures

$$(\mathcal{P}X, \cup, \emptyset) \quad \text{and} \quad (\mathcal{P}X, \cap, X)$$

but it seemed that only the former was the most useful in 'practical' specifications. All of the other operations on sets were of course available and used. It was only very recently that the real significance of the second monoid $(\mathcal{P}X, \cap, X)$ became clear in its role for practical specifications. First I thought that the proper structure ought to be the Boolean lattice. Finally, I recognized that it was most appropriate to use the Heyting algebra [Mac an Airchinnigh 2001]. With it one could do logic algebraically.

For any set X one could apply the star operator to obtain the sequences of elements of X, denoted $X^\star$. Again it was always assumed that X would be finite in practice. The basic structure was the free monoid

$$(X^\star, \cdot, 1)$$

where $\cdot$ denotes concatenation. Sequences were deemed to be the essential abstraction for the concept of "implementation on a computer".

The most important model was provided by the collection of partial (and total) maps from X to Y, denoted $X \longrightarrow Y$. The basic monoid was

$$(X \longrightarrow Y, \dagger, \theta)$$

where $\dagger$ denoted override and θ was the empty map. Composition of maps is the normal operation in mathematics. Override of maps is peculiar to computer science and override is especially significant in the context of partial maps.

At any given time one might consider one such map, say μ. It is my recollection that the convention adopted to denote this was to name the collection of maps, say M, and to write

$$\mu \in M = X \longrightarrow Y.$$

I am almost certain that I picked up this convention from Andrzej Blikle at one of those early Brussels' meetings. *However, this convention is to be abandoned even as I write.* I now have an urgent need to integrate the VDM$^\clubsuit$ fully with topos theory and an expression such as $X \longrightarrow Y$ is already well established to denote one total map from

X to Y in the topos of sets. I am also happy to use $X - - \succ Y$ to denote a strictly partial map from X to Y. One can think of it as shorthand for the pair of total maps

$$X \xleftarrow{\ i\ } \mathsf{dom}\,\mu \xrightarrow{\ \mu\ } Y.$$

How then ought one to denote the collection of maps? I certainly can no longer use $X \longrightarrow Y$. The simplest solution is to use Y^X for the collection of total maps and $\tilde{Y}^X$ for both partial and total maps, where $\tilde{Y} = Y + \mathbf{1}$ in the topos of sets. The notation is taken directly from that used in topos theory. I must also say goodbye to the 'Blikle convention' at least in the introductory work. (It can be completely rescued by an appropriate definition of 'membership'.) In whatever topos one finds oneself one can always pick out a map μ by writing $\mu\colon \mathbf{1} \longrightarrow \tilde{Y}^X$ or

$$\mathbf{1} \xrightarrow{\ \mu\ } \tilde{Y}^X$$

But in my opinion this is not completely psychologically satisfying from the desire to handle partiality directly as we are wont to do in the VDM$^{\clubsuit}$.

Relations were treated as set-valued maps. I adopted this (philosophical) position from the very beginning, following [Eilenberg 1974], one of the two founders of Category Theory. (The other was Saunders Mac Lane).

$$X \longrightarrow \mathcal{P}Y$$

For example, the usual model of doctors (DOC) and their patients (PAT) that I had become accustomed to use is that which associates with each doctor d in the klinik κ her/his set of current patients S. This is the classical doctor-patient relationship. This model is captured by

$$\kappa \in \mathit{KLINIK} = \mathit{DOC} \longrightarrow \mathcal{P}\mathit{PAT}$$

and a typical klinik κ might have the form

$$\kappa = \begin{bmatrix} c \mapsto \{p, q, r\} \\ d \mapsto \{p, s\} \\ e \mapsto \emptyset \end{bmatrix}$$

It will be noticed that in this model the same patient p might be shared between two doctors c and d, and there is a doctor e with no patients.

This model of a klinik was considered to be the most general abstract model of the doctor-patient relation. It is a *directed* model in the sense that the relation is "the doctor d has the set of patients S".

However, this is definitely not the end of the story of relations in the M$^{\clubsuit}_{\mathsf{C}}$. In the Berlin paper [Mac an Airchinnigh 2001] I gave another map-oriented approach to relations. There is yet a further development in the use of maps to model relations. Unfortunately, this will have to wait until the next paper/tutorial. Suffice it to say that I still

adhere to the principle of handling relations as maps, though not limited to set-valued ones. Fortunately, the discovery of the topos and its logic fits in with that view very nicely.

Finally, distributions, indexed structures, etc. took the form

$$X \longrightarrow (Y \longrightarrow Z)$$

It was quite clear that one could model anything by putting together these mathematical structures. The result was not always very elegant. Things are better now and more elegant, 10 years later.

In the method of the School presented in Noordwijkerhout I also presented a variety of development (reification or development or elaboration) steps: partitioning (or subdividing), splitting, parameterising, and joining. The interested reader is directed to [Mac an Airchinnigh 1991]. Proofs relied upon the use of commuting diagrams which I took directly from category theory.

Many important developments of the School took place in the intervening years from 1991 (Noordwijkerhout) to 2001 (Berlin). The best public record and summary of these is [Mac an Airchinnigh 2001]. Now I should like to introduce the second part of the paper. It consists largely of a very elementary introduction to topos logic. However, before jumping straight in, it is necessary to put on record, the primacy which constructive mathematics will occupy in computer science for the foreseeable future. Like everyone else in Mathematics, I am a constructive mathematician from Monday to Friday and a Neoplatonic mathematician at the week-end.

3 Intensional Reality

> "*First we want to distinguish between **functions** and **forms.*** $2 \times x + y$ *is a form containing **free variables*** x *and* y*. Implicitly by convention or explicitly, each variable is associated with a domain of values. A map from the free variables of a form to their value domain is called a value assignment. A form together with a **value assignment** to all its free variables denotes a value. To account for undefined situations the value domain has to include an error element or the 'meaning function' is defined to be a partial function.*" [Lucas 1985] p. 147.

In 1985 I might have agreed with Peter Lucas' distinction between functions and forms. By 1990 when I finished my doctoral thesis, *Conceptual Models and Computing* [Mac an Airchinnigh 1990] I disagreed!

In trying to establish the School of Constructive Mathematics, I wanted to avoid any 'taint' with classical intuitionistic mathematics. I knew that the mathematics of computing was de facto constructive mathematics. That much was self-evident. That a formal method was essentially the mathematics of computing was also obvious. That most formal methodists used classical logic and pooh-poohed intuitionistic logic was

clear. I was not comfortable with any logic, classical or intuitionistic! Then I discovered Topos Theory.

One is so conditioned by prior education to think of maps extensionally that even when one is fully aware of the intrinsic intensional nature of maps in computing one still resorts to extensional confession. This conditioning is effectively a Neoplatonic hangover from the pre-computing era.

Let me give the example that I normally use in class. Consider the maps $\mathbb{R} \xrightarrow{f} \mathbb{R}$ and $\mathbb{R} \xrightarrow{g} \mathbb{R}$ where

$$f\colon x \mapsto (x+1)^2$$

and

$$g\colon x \mapsto x^2 + 2x + 1$$

Were one to graph the two maps (using Mathematica [Wolfram 1996] for example) then it would be immediately obvious that both f and g were the same map in the sense that given an input x one would get exactly the same output $f(x) = g(x)$.

To be more precise one would say that extensionally we have a map defined uniquely by the set of pairs $\langle\, x, y \,\rangle$ where y is computed by $f(x)$ or equally by $g(x)$, i.e., one refers to the unique map

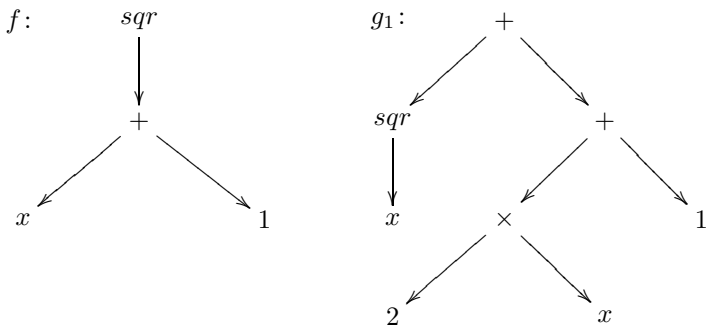
$$\{\langle\, x, y \,\rangle \mid y = f(x)\}$$

Such a Neoplatonic perspective may be termed the *catholic* view (where catholic is taken to mean universal). There is an equally valid alternative *mathematical* view, especially significant for computer scientists. In contradistinction to the from-heaven-descending Neoplatonic view of the ideal, *the given*, there is the more obvious from-earth-ascending Neo-aristotelian view of the obvious becoming, the process. In short to our minds f and g are distinct. Thanks eventually to the *Elements of Intuitionism* of [Dummett 2000] I can now comfortably take hold of and align myself with a philosophically acceptable position on intuitionism and be freed from any direct reliance upon Brouwer, though must acknowledge the direct influence of Heyting:

> "*The description by which a mathematical object is given must always be such as to enable it to be distinguished from other objects of the same kind. However, since mathematical objects are mental constructions, and the mental construction is expressed by means of the description in terms of which the object is given, the objects of intuitionistic mathematics must, in general, be considered as intensional objects; that is to say, that criterion of identity which is given together with the manner in which the object is presented relates to the identity of the description. Thus, for example, if an effectively calculable function is thought of as given by means of a rule of computation, different rules will*

determine intensionaly distinct functions, even if these functions are extensionally equivalent, i.e. have the same values for the same arguments." [Dummett 2000] p. 17.

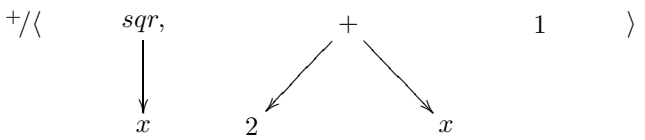
Let us apply this criterion of identification and description to the two maps f and g given above. As computer scientists we know that maps which are defined using algebraic formulae are determined intrinsically by both a parse tree and an evaluation mechanism or process. This is the Lucas' view. For example, the parse tree of f is uniquely determined; that of g is ambiguous. The formula may be written either as $x^2+(2\times x+1)$ or as $(x^2+2\times x)+1$. My choice of the former is designated g_1. In addition to the parse tree structure there is the evaluation mechanism. Again we computer scientists are familiar with three classical canonical evaluation mechanisms for binary parse trees: prefix, infix, and postfix. More significantly, these evaluation mechanisms may be given very precisely (i.e. formally as recursive algorithms over structure; there is also the possibility of the more general probabilistic algorithm over structure.



Since computer scientists know these distinct conceptual constructive descriptions then according to the fundamental principle of intuitionistic mathematics these are objects of the same kind but presented differently. These are intensionally different objects. For example, and it seems to me now to be a trivial remark, I can define a simple complexity function which highlights the extraordinary richness of the formally given description of the intensional objects. Specifically, by adopting the *binary parse tree* conceptual framework as part of our description language, I can immediately speak of a height (or depth or level) function which gives the height of the tree. Thus, for f the height is $h_f = 2$. For g_1 the height is $h_{g_1} = 3$. In a sense, g_1 is more complex than f. (Note that I define $sqr(x) = x \times x$.) Let us then define the 'cost' of addition to be a units and the cost of multiplication to be m units (say $m = 10a$). This enriches in a most extraordinary way the intensionality of the concept of function. Specifically the conceptual framework becomes increasingly more relevant. In other words, the speed of the processor in computing additions and multiplications (et cetera) becomes conceptually significant. Using this notion of a cost function then the cost of computing f *independent of evaluation process* is $c_f = m + a$; the cost of computing g_1 is $c_{g_1} = 2m + 2a$. Not only is the cost of computing g_1 twice that of computing f but also the structure

(i.e. height of parse tree) of g_1 is more complex than that of f. Furthermore, computer scientists know that there is a cost associated with parse trees, a storage cost. Hence in the determination of the conceptual understanding of the presentations of the functions f and g there is another cost function—the storage function. It is no wonder that the mathematics of computing, constructive or intuitionistic mathematics is so very different from the usual classical and (mostly) irrelevant mathematics of the natural sciences!

Once freed from the extensional constriction one can explore other intensional forms other than the singly-rooted parse trees given above:



In particular, any polynomial expression such as $x^2 + 2x + 1$ is quickly associated with the reduction $^+/\langle\, x^2, 2x, 1 \,\rangle$. Such a reduction is simply another way of representing, in this case, a summation $\sum_{k=1}^{3} \sigma_k$ where σ is the vector (i.e., sequence) of three polynomial expressions. Such a reduction is naturally associated with parallel execution, at least conceptually. In this case we have three parallel trees to be evaluated and the final result is a summation of these.

To say that one confuses function with algorithm is an argument whose time is long past! The framework of topos theory with its associated logic exhibits in a most profound manner that not only is the intensional world *at least on a par with* the extensional one given to us by Cantor, but is in fact much richer. On the other hand there does not seem to be any way in which to exhibit this richness without going through topos theory itself. This is the task at hand. This is the subject matter for the remainder of the paper.

4 Logic for VDM♣ via Topoi

> "*There will be much [discussion] about the analysis that goes into deciding what need[s] to be done, and in what order. Anyone who has struggled with a genuine problem without having been taught an explicit method knows that this is the hardest part.* [slightly paraphrased by the author] " [Lawvere and Schanuel 1997] p. xiii

In August 2000 I reached the decision that the time was ripe for the next international tutorial of the VDM♣. It was called "Modelling for Formal Methods" and was delivered at FME 2001 in Humboldt-Universität zu Berlin on the 12th of March 2001. In essence it was a one day personal exposition and view of *Conceptual Mathematics* [Lawvere and Schanuel 1997] seen from the practical perspective of our School. The target audience was "Anyone interested in the learning how to apply category theory and topos theory for the purposes of abstract modelling." The morning

session covered the *highlights* of category theory, the afternoon session those of topos theory, both with an emphasis on modelling. In attendance was Peter Lucas.

Since March 2001 I have condensed this mini-tutorial to 45 minutes. Since the School had just spent an intensive year molding and fashioning topos theory to suit its needs and since the topos carried inherent programming structure/process together with the corresponding inseparable logic I realized that it was possible to introduce topos theory directly without having to wade through an enormous amount of 'categorical preliminaries'. For an audience of computer scientists, a direct assault on topos logic was feasible. Therefore, a 45 minute delivery was also feasible. Peter Lucas and friends and colleagues and students were all subjected to this at his Colloquium on Friday, the 18th of May in Graz, Austria. The following is a brief description of the technical material covered.

I assume that the reader is at least familiar with the most basic concept of a category. (If not, I recommend that the reader do a thorough study of *Conceptual Mathematics* [Lawvere and Schanuel 1997]. Proceed as follows. Start at the beginning, i.e. the **front cover**, and read right through to the end. Do not wait to study. Do not do exercises. Note the things that surprise. Then start at the beginning again with an open mind.) Specifically, one has a collection of things called objects. There is also a collection of things called maps (or arrows or ...) which may be combined under a law of composition. For each map $f\colon A \to B$, we say that the domain of f is the object A (i.e. object A is the source of the map) and that the object B is the codomain of the map (i.e. object B is the target of the map). The composition of maps $f\colon A \to B$, and $g\colon B \to C$, is the map $gf\colon A \to C$, read "g after f" or "g of f". Composition of maps satisfies the associative law. For each object X there is an identity map $1_X\colon X \to X$. Finally, there are certain "housekeeping rules" which must be observed.

The importance of the category lies in the simple observation that it provides the machinery for conceptual mathematics, that is to say, it provides the mathematics that facilitates the formalization of thought, in general.

Were one to ask what is the essence of computing, then one might provide a great variety of different answers. A simple yet inclusive answer is to reply by one word: *programming*. Of all the programming language paradigms currently in existence, surely that of functional programming is closest to that of the conceptual mathematics of categories? Without wishing to be overly simplistic I can say that functional programming is essentially founded on the Cartesian Closed Category (abbreviated CCC). The quickest way to grasp CCC is to think of the programming language Pascal as an overt and direct realization/manifestation of it. By this I mean that every key construct in Pascal corresponds directly to a key aspect of the CCC. (I am sure that there must be a good reference for this claim somewhere.) Now there remains but one more essential concept, that of **locality** or of '**part-ness**'. For to know whether one object is a part of another, and to classify it as such, one introduces the concept of the truth value object which provides the basis for an inherent 'intuitionistic' logic. Essentially the CCC to-

gether with the truth value object gives us a very special type of category which we call a topos. (For details see [Lawvere and Schanuel 1997].) Its significance lies in the fact that not only can we be precise about what we can compute but also that we can reason and make judgments about what we compute locally. The very nature of the topos and, in particular its locality facet, lends itself pre-eminently to the modelling of both *distributed systems*, in general, and *mobility* in particular. From the perspective of the mathematician, the topos provides a setting for the *unification of algebra, geometry, and logic*. (See [Mac Lane and Moerdijk 1992].)

It has long been known that mathematics is the foundation of science as we know it. Mathematicians know that in many respects mathematics is like poetry. It is a different way of knowing and manifesting the world. Let us look at some simple notions concerning **citizenship** and **information**. Now I would like to demonstrate that it is indeed possible to use mathematics, more specifically the conceptual mathematics of topos theory, to model the notions of citizenship and information and to reason in a precise manner about their inter-relationships. Undeniably, in order that one may feel at home in such a mathematical framework, one needs to be familiar with the language. It does take time to achieve fluency and familiarity. Like any language, it takes some years. Yet, I trust that this part of the paper might encourage some readers to make that effort and so come to know a little of the conceptual mathematics that will be taken for granted in our future.

Let us then begin. In the first instance let us assume that we are in the usual mathematical world (i.e. topos) of sets and total maps (denoted $\mathcal{S}$). We let upper-case letters, A, B, $\ldots$, X, Y, denote sets. We let lower-case letters, a, b, $\ldots$, f, g, h, $\ldots$, x, y, z, denote elements of a set and total maps. A map f from X to A is usually written $f\colon A \to B$. Let X be a set of, say, three colours, r, g, and b (representing red, green, and blue, respectively). We customarily denote this by $X = \{r, g, b\}$. Now let us choose a set of one element $\{*\}$ which we shall denote by $\mathbf{1}$. Then it is clear that there is exactly one map from X to $\mathbf{1}$. In computer science, we might consider this singleton set $\mathbf{1}$ to represent the end point or sink state or terminal state of a development or process. From the point of view of X, the unique map $!_X\colon X \to \mathbf{1}$ takes every element of X to $*$. We can make this explicit by writing

$$!_X = \begin{bmatrix} r \mapsto * \\ g \mapsto * \\ b \mapsto * \end{bmatrix}.$$

It is customary to call $\mathbf{1}$ the terminal object. I often think of it as **the unique one**! There is also the unique identity map $\mathbf{1} \to \mathbf{1}$. From a modelling perspective we might prefer something a little bit more concrete and applicable to the 'real' world.

Let us suppose that we have some non-empty set A which models a collection of students in a class. Then the 'unique one' will be deemed to be the **focus of attention** of the collection. To model this we introduce a map from the collection A to the $\mathbf{1}$. This abstraction captures quite nicely the relationship between a class of students (A) and

their lecturer ($\mathbf{1}$) or a flock of sheep (A) and their shepherd ($\mathbf{1}$). We call the focus of attention of A on the $\mathbf{1}$ a constant map where by constant we mean unchanging and unchangeable.

Returning to the colours, now let us put ourselves in the place of the terminal $\mathbf{1}$ and look back at the colours X. What is it that we see? If we model 'seeing' by a map then there are three distinct maps which we shall denote by $r\colon \mathbf{1} \to X$, $g\colon \mathbf{1} \to X$, and $b\colon \mathbf{1} \to X$. Quite obviously, these three maps **point** out the three colours of X. This is the reason that we name the maps (and the colours) r, g, and b. In general, it is always the case (in the category of sets $\mathcal{S}$) that a map $\mathbf{1} \to A$ picks out, or points to, an element of A (provided, of course that A is not empty). Formally, in this mathematical world (i.e. topos) we define any such map $\mathbf{1} \to A$ to be a point.

DEFINITION 1 A point of a set X is a map $\mathbf{1} \to X$ [Lawvere and Schanuel 1997] p. 19.

Armed with the terminal $\mathbf{1}$ and the definition of point as map we can now construct (i.e. model) some very interesting concepts in this world of sets and total maps. For example, suppose that we imagine a set C to represent the citizens of a particular region. Then every point of C, denoted by the map $\mathbf{1} \to C$ is a citizen of C and the number of such points gives the number of citizens. Would we not like to be able to distinguish between citizens and non-citizens of a region? In other words if we ask the question "is so and so a citizen of this region?", then we expect to get an answer such as yes or no. How might we model that?

First, we introduce a two point set. Each point will represent a truth value. This two point set will be denoted by Ω and the points by $t\colon \mathbf{1} \to \Omega$, and $f\colon \mathbf{1} \to \Omega$, representing true and false, respectively. It is usually called the truth value object. (Other notations in common use for the Ω of the topos of sets are $\mathbb{B}$ (for Boolean) and **2** (for two-point). To be more precise we want a map to model what it means for something to be true. Once we have decided on this then we also *de facto* have determined what it means for something not to be true (i.e. to be *not*-true, i.e. to be false).

Now let us look at the maps from Ω to itself. There are exactly $|\Omega|^{|\Omega|} = 2^2 = 4$ such maps. The identity map, which always exists between a set and itself, is here denoted 1_Ω . Of the other three maps, the 'inversion' map, which we usually call negation and denote by $\neg\colon \Omega \to \Omega$, is especially noteworthy. It takes t to f and f to t. In other words we have the compositions $\neg t\colon \mathbf{1} \to \Omega \to \Omega$, and $\neg f\colon \mathbf{1} \to \Omega \to \Omega$, with $\neg t = f$ and $\neg f = t$. The other two maps are constant maps which we will denote by T and F, respectively. They are defined in the obvious way: $Tt = t$, $Tf = t$ and $Ft = f$, $Ff = f$.

In general, for any two non-empty sets A and B a map between them, say $A \to B$, can be understood in terms of *structuring* either the domain A or the codomain B [Lawvere and Schanuel 1997] p. 81. Let us look at domain-structuring maps. Consider the map $g\colon X \to B$. We will say that g is a sorting or fibering of X by B into B *sorts* or

B fibers, respectively. (The language used evokes 'application domains' and therefore 'modelling abstractions'. The word sort suggests 'kind' and 'type'. There is a famous brand of sweets called 'liquorice all sorts', meaning all sorts of liquorice. The word fiber evokes agricultural terms used very successfully by the French mathematicians in geometry.)

Let us consider a simple example. If X were to denote a set of people (i.e. citizens and non-citizens) in a given region and B were to denote the truth value object Ω, then there is a map g which takes the citizens of X to the truth value t in Ω. We call such a map g a (subobject) classifier. As a fibering, g structures X into exactly two sorts or fibers, one of which we identify as the sort of citizens, precisely because every point of the fiber maps to t. There yet remains the problem as to how we can pick out the citizens in X. Clearly, if we have this classifier g, then the inverse image or pullback g^{-1} of t are the citizens.

On the other hand, consider a map $f\colon A \to X$ which produces structure in the codomain. We think of the image of A in X under f as an A-shaped figure. Other ways of viewing the structuring of the codomain is to think of f as naming the elements of X or even as a sampling or parameterizing of X by A. Now let us introduce the set of citizens C and an inclusion map (which is by definition an injective map or one-to-one map) $C \overset{i}{\hookrightarrow} X$ which identifies the citizen part of X, that is to say the C-shape of X is exactly that part (or subset) of X which we want to classify as the citizens. More generally,

DEFINITION 2 In any category, a map $S \overset{i}{\hookrightarrow} X$ is an inclusion, or monomorphism, or monic, if it satisfies: For each object T and each pair of maps s_1, s_2 from T to S, $is_1 = is_2$ implies $s_1 = s_2$ [Lawvere and Schanuel 1997] p. 336.

Finally, the composition $gi\colon C \to X \to \Omega$ takes every citizen $c\colon \mathbf{1} \to C$ to $t\colon \mathbf{1} \to \Omega$. We can be even more precise about the way in which we denote the exact relationship between that part of X which we pick out as C and our judgment to classify the part as denoting citizens. Specifically, instead of using an arbitrary letter, such as g we recognize that our choice is and must be directly related to C. Two common conventions are φ_C and χ_C. In other words, the classifying map corresponds exactly to the part being classified. We capture the totality of this relationship by saying that we have the natural bijection

$$\frac{C \overset{i}{\hookrightarrow} X}{X \xrightarrow[\varphi_C]{} \Omega}$$

This means that corresponding to each map $C \overset{i}{\hookrightarrow} X$ (which is a subset inclusion and which one might write in set notation as $C \subseteq X$) there is a map, $X \xrightarrow{\varphi_C} \Omega$, called the classifying map or characteristic map, indexed by C, which classifies this subset, and *vice versa*.

This classifying of the citizens automatically classifies the non-citizens at the same time. Furthermore we say that not C is *the largest part of X which is disjoint from* C. This is our (intuitively correct) definition of what not means with respect to part-ness. The next obvious question is how this sense of not fits in with the notion of negation. Given what we have done so far it should not be difficult for the reader to demonstrate that in this topos

$$\varphi_{\mathsf{not}\ C} = \neg \circ \varphi_C, \quad \text{abbreviated to } \neg\varphi_C$$

Clearly not not C is exactly C. Correspondingly, $\neg^2 = \mathbf{1}_\Omega$. Our world of sets and total maps gives us the classical two valued (Boolean) logic. In particular, there can never be any doubt about whether or not one is a citizen. [Aside: A more comprehensive understanding of the relationship between not and negation is outlined in [Lawvere and Schanuel 1997] p. 350–1. I do not explore this here now for the simple reason that I do not yet have a satisfactory intuitive way of explaining the *relationship* between 'logical implication' $\Omega \times \Omega \xrightarrow{\Rightarrow} \Omega$ and part-ness. It is work in progress.]

Let us now consider 'information'. To be more precise we will deal with an information resource. Such a resource is a well-defined and easily recognised entity which can be named and pointed to. Such a resource will be deemed to consist of a collection of information units organized in some fashion and inter-related (e.g. cross-referenced). We shall model an information resource in a graph theoretic manner. Specifically, each vertex m of a graph shall represent a specific unit of information which might be the page of a book, a World-wide Web (www) page, a resource person, etc.

A directed edge e from vertex m to vertex n shall be used to model a reference, a citation, a link, etc. from one information unit to another information unit. Clearly, there will be cases when an information unit might reference itself. For simplicity, we will refer to the vertices as dots and to the edges as arrows. Our vision of the model of an information resource is a directed multi-graph. In itself it is an object which consists of a set of dots D, a set of arrows A and two maps from A to D called the source and target maps, denoted $s\colon A \to D$ and $t\colon A \to D$, respectively. The corresponding topos of directed multi-graphs we denote by $\mathcal{G}$. A simple example shall make this clear. Suppose that a particular information resource consists of a collection of interlinked www pages numbered 0, 1, and 2. The links between the three information units are given by arrows (in triplet form):

$$\langle\, 0, a, 1 \,\rangle, \langle\, 1, b, 1 \,\rangle, \langle\, 1, c, 2 \,\rangle, \quad \text{and} \quad \langle\, 2, d, 2 \,\rangle,$$

where the triple $\langle\, x, e, y \,\rangle$ designates an arrow $x \xrightarrow{e} y$, named e which has source x and target y. (See [Fig. 1] below.) Thus, for example, page 1 has a self link named b and a link named c to page 2 of the collection of information units. Formally, this information resource which consists of three interlinked pages is an object G determined

fully by the following data:

$$
\begin{aligned}
A_G &= \{\langle\, 0, a, 1 \,\rangle, \langle\, 1, b, 1 \,\rangle, \langle\, 1, c, 2 \,\rangle, \langle\, 2, d, 2 \,\rangle\} \\
D_G &= \{0, 1, 2\} \\
s &= \begin{bmatrix} \langle\, 0, a, 1 \,\rangle \mapsto 0 \\ \langle\, 1, b, 1 \,\rangle \mapsto 1 \\ \langle\, 1, c, 2 \,\rangle \mapsto 1 \\ \langle\, 2, d, 2 \,\rangle \mapsto 2 \end{bmatrix} \\
t &= \begin{bmatrix} \langle\, 0, a, 1 \,\rangle \mapsto 1 \\ \langle\, 1, b, 1 \,\rangle \mapsto 1 \\ \langle\, 1, c, 2 \,\rangle \mapsto 2 \\ \langle\, 2, d, 2 \,\rangle \mapsto 2 \end{bmatrix}
\end{aligned}
$$

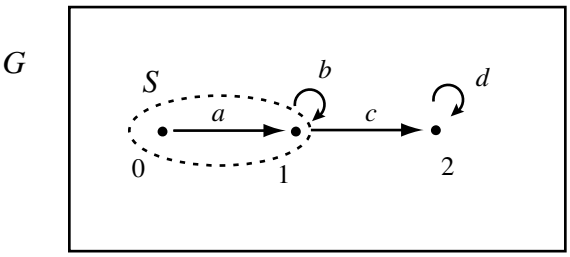


Figure 1 The object G.

It is customary to represent this object by a diagram such as that shown above in Figure 1. Note that a part of the object G, named S, is indicated on the diagram by a dashed oval. The data for S is clearly

$$
\begin{aligned}
A_S &= \{\langle\, 0, a, 1 \,\rangle\} \\
D_S &= \{0, 1\} \\
s &= [\langle\, 0, a, 1 \,\rangle \mapsto 0] \\
t &= [\langle\, 0, a, 1 \,\rangle \mapsto 1]
\end{aligned}
$$

To say that S is a part of G one must also say that there is an inclusion map $S \overset{i}{\hookrightarrow} G$. Thus, it is more precise to say that S, i is a part of G. Alternatively, if one is aware that the inclusion map i is furnished with domain S and codomain G then the map i says it all [Lawvere and Schanuel 1997] p. 338.

In this world the terminal object $\mathbf{1}$ is a graph which has exactly one dot and a self-loop. The typical picture looks like

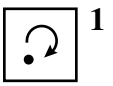
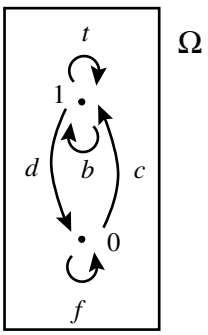


Figure 2 the terminal graph object.

By definition a point must have the shape of the terminal. Hence there are just two points in G: $\langle\, 1, b, 1 \,\rangle$ and $\langle\, 2, d, 2 \,\rangle$. From a modelling perspective, self-references in information units are points. The truth value object Ω for directed multi-graphs is given by the following data [Lawvere and Schanuel 1997] p. 341:

$$
\begin{aligned}
A_\Omega &= \{\langle\, 1, t, 1 \,\rangle, \langle\, 1, b, 1 \,\rangle, \langle\, 1, d, 0 \,\rangle, \langle\, 0, f, 0 \,\rangle, \langle\, 0, c, 1 \,\rangle\} \\
D_\Omega &= \{0, 1\} \\
s &= \begin{bmatrix} \langle\, 1, t, 1 \,\rangle \mapsto 1 \\ \langle\, 1, b, 1 \,\rangle \mapsto 1 \\ \langle\, 1, d, 0 \,\rangle \mapsto 1 \\ \langle\, 0, f, 0 \,\rangle \mapsto 0 \\ \langle\, 0, c, 1 \,\rangle \mapsto 0 \end{bmatrix} \\
t &= \begin{bmatrix} \langle\, 1, t, 1 \,\rangle \mapsto 1 \\ \langle\, 1, b, 1 \,\rangle \mapsto 1 \\ \langle\, 1, d, 0 \,\rangle \mapsto 0 \\ \langle\, 0, f, 0 \,\rangle \mapsto 0 \\ \langle\, 0, c, 1 \,\rangle \mapsto 1 \end{bmatrix}
\end{aligned}
$$



Ω

Figure 3 The truth value object for graphs.

A typical picture of the truth value object is shown opposite in Figure 3.

This encodes seven possible truth relations (or statements) that one can make about the subgraph of a graph [Lawvere and Schanuel 1997] p. 341:

1. $\langle\, 1, t, 1 \,\rangle$: arrow in; source in; target in;
2. $\langle\, 1, b, 1 \,\rangle$: arrow out; source in; target in;
3. $\langle\, 0, f, 0 \,\rangle$: arrow out; source out; target out;
4. $\langle\, 1, d, 0 \,\rangle$: arrow out; source in; target out;
5. $\langle\, 0, c, 1 \,\rangle$: arrow out; source out; target in;
6. 1 : dot in;
7. 0 : dot out.

Note that the labelling of dots and arrows is local to the object in question. In particular, $D_G = \{0, 1, 2\}$ and $D_\Omega = \{0, 1\}$ have nothing in common!

Moreover, it also clear that there are only three truth values in the truth value object Ω. These are the points named by the arrow t (for true), the arrow f (for false) and the arrow b (for in-between, i.e. neither true nor false). Now there is a subobject classifier map $g\colon G \to \Omega$ which takes the object S to true (i.e., which classifies S, i to be a part of G). The rest is classified appropriately. For example, the arrow $\langle\, 2, d, 2 \,\rangle$ in G is mapped to $\langle\, 0, f, 0 \,\rangle$ (i.e. false) in Ω since $\langle\, 2, d, 2 \,\rangle$ is not in S. We define not S to

be the largest part of G which is disjoint from S [Lawvere and Schanuel 1997] p. 351. We illustrate this in Figure 4.

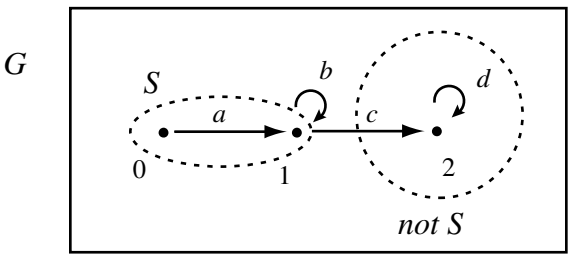


Figure 4 not S.

Since the formal data of a graph is given in terms of two sets, and two maps, then we can calculate not S quite simply. The construction is as follows. First we observe that the dots play a crucial role and $D_{\mathsf{not}\ S}$ should be calculated first:

$$D_{\mathsf{not}\ S} := D_G \backslash D_S$$

Hence

$$D_{\mathsf{not}\ S} = \{0, 1, 2\} \backslash \{0, 1\} = \{2\}$$

The only arrows of A_G which have source 2 and target 2 is $\langle\, 2, d, 2 \,\rangle$. Hence,

$$A_{\mathsf{not}\ S} = \{\langle\, 2, d, 2 \,\rangle\}$$

Now let us calculate $T = \mathsf{not\ not}\ S$. Pictorially, the result is shown below in Figure 5. Again we can calculate the formal data for the subobject not not S, beginning necessarily with the dots.

$$D_{\mathsf{not\ not}\ S} := D_G \backslash D_{\mathsf{not}\ S}$$

Hence

$$D_{\mathsf{not\ not}\ S} = \{0, 1, 2\} \backslash \{2\} = \{0, 1\}.$$

Then $A_{\mathsf{not\ not}\ S}$ consists of all the arrows which have 0 and 1 as source and target dots. Hence

$$A_{\mathsf{not\ not}\ S} = \{\langle\, 0, a, 1 \,\rangle, \langle\, 1, b, 1 \,\rangle\}.$$

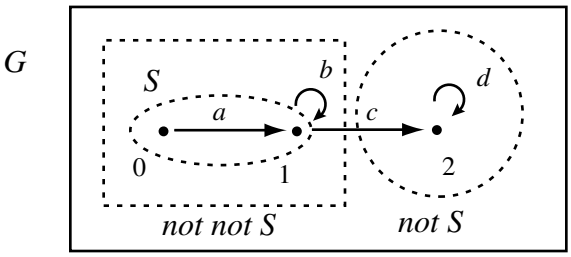


Figure 5 not not S.

From the diagram we observe two very important things, typical of intuitionistic logic [Dummett 2000]. First, it is clear that S and not S do not re-constitute G. Second, S is strictly contained within not not S. From a practical point of view let us suppose that we have an information resource modelled by G. Then if we focus on that part of G which we have named S and ask what is in G which is not modelled by S? The answer must be given relative to G; it is that part which is a self-referencing page 2. There can be no notion of a reference from page 1 to page 2. In effect not S identifies the largest information resource in G which is independent of S. That is a very useful and practical result. In short, suppose that we have identified S as being un-trustworthy. Then we should concentrate our attention on not S in the hope that we can retrieve some trustworthy information from G.

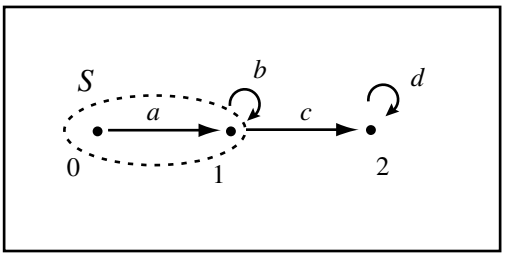
Now, focusing on page 2 (i.e. not S), we ask what is in G which is not in not S we obtain pages 0 and 1 together with their own self contained references. Again there is a practical outcome! Suppose that we have identified S as being trustworthy. Now we seek to maximise that subpart of G which is trustworthy. By constructing not S and interpreting it as potentially un-trustworthy then not not S gives the maximal subobject containing S in G which can be self-contained and, therefore, potentially independent and trustworthy. Now let us ask what is the boundary of all the possible maximal subobjects not not S for any subobject S with two pages (i.e. two dots)? The answer is clear: any maximal subobject must be part of (including the possibility of 'being equal to' in the limit) the complete multi-graph on two nodes. If we allow an arbitrary number of edges between nodes then this complete graph is potentially infinite. In a practical design one will wish to impose a limit on the number of edges.

With this particular observation on the practicality of intuitionistic logic for information resources and their trustworthiness we must draw the modelling section to a close. Let us sum up the previous discussion by presenting the classifiers in a VDM map style alongside the actual graphs:

classifier of S wrt G

$$\varphi_S = \begin{bmatrix} a \mapsto t \\ b \mapsto b \\ c \mapsto d \\ d \mapsto f \end{bmatrix}$$

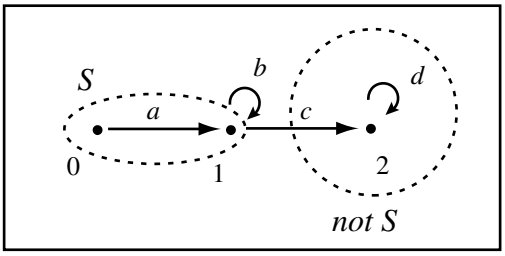
G



classifier of not S wrt G

$$\varphi_{\text{not } S} = \begin{bmatrix} a \mapsto f \\ b \mapsto f \\ c \mapsto r \\ d \mapsto t \end{bmatrix}$$

G

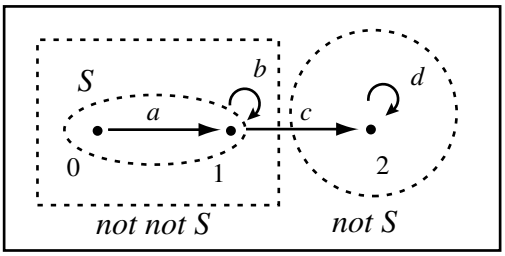


In general, $S \cup \text{not } S \subseteq X$, where $\cup$ is used to denote the 'unions' of sets of arrows and sets of dots, and the respective source and target maps. It is an interesting exercise to write this out in detail in VDM$^{\clubsuit}$ notation. In this particular case, one immediately observes that $S \cup \text{not } S \neq X$.

classifier of not not S

$$\varphi_{\text{not not } S} = \begin{bmatrix} a \mapsto t \\ b \mapsto t \\ c \mapsto d \\ d \mapsto f \end{bmatrix}$$

G



In general, $S \subseteq \text{not not } S$. In this particular case, one immediately observes that $S \neq \text{not not } S$.

Finally, the reader is invited to demonstrate that in the topos of graphs $\neg \circ \neg \neq 1_\Omega$.

$$\neg = \begin{bmatrix} t \mapsto f \\ b \mapsto f \\ f \mapsto t \\ d \mapsto r \\ r \mapsto d \\ 1 \mapsto 0 \\ 0 \mapsto 1 \end{bmatrix}$$

The logic of the topos of graphs is not the classical two valued logic. It is, in fact a three valued logic, as seen from the outside, from the position of an external viewer. We deduce this fact by counting the number of points. Internally, there is a richness of levels of truth.

From the perspective of the computer scientist, the result is startling. We are very accustomed to use graph theory to model all sorts of things such as communication networks, flow graphs of programs, etc. The natural reasoning that underlies this is founded on intuitionistic logic. We have just demonstrated that reasoning with the latter seems to be a little bit more complicated than reasoning with classical logic. In a topos theoretic framework both reasoning processes are 'identical'. The difference depends on the shape of the topos. The $M_C^{\clubsuit}$ can cope with both equally well. Our future looks bright!

5 Epilogue

> "*A startling aspect of topos theory is that it unifies two seemingly wholly distinct mathematical subjects: on the one hand, topology and algebraic geometry, and on the other hand, logic and set theory. Indeed, a topos can be considered both as a 'generalized space' and as a 'generalized universe of sets'.*"
> [Mac Lane and Moerdijk 1992] p. 1.

The challenge of the $M_C^{\clubsuit}$ is to make topos theory accessible to all who are interested in the evolution of modelling and specification and to turn it into a very practical mathematical tool for computer science. The [Lawvere and Schanuel 1997] work, *Conceptual Mathematics*, is a landmark publication. The only thing missing from it is the introduction of the override operator for maps. In a recent doctoral thesis in the $M_C^{\clubsuit}$, *Elements of an Operator Calculus* [Hughes 2001] has taken the first step to remedy this omission by giving the semantics of override in several different topoi. Our future in the School looks very bright indeed.

However, I must confess that I think Peter Lucas is not yet convinced of the practicality of our topos theoretic approach. Such, I believe was his opinion in Berlin (2001) during our inaugural tutorial on the subject. Indeed, he is quite correct. There is still a lot of work to be done, mostly at the pedagogical level. It is time to draw this paper to a close.

I brought my old 1978 LNCS **61** with me to the Colloquium in Graz. It is now physically (and topologically) separated into about 23 different parts, many of which are single-leaved, and is customarily held together on the bookshelf by a large elastic band. On the 18th of May I obtained, at last, the signatures of Cliff Jones and Peter Lucas, both dated, Cliff's dated using the ISO date form 2001-05-18, which form I always use myself. I am impressed. The only contributor to the volume whose signature I have not yet got and whom I have never yet met is that of Wolfgang Henhapl. I have not given up hope.

LNCS **61** is for me a very valuable book. I used always to think of it in terms of [Bjørner and Jones 1978] . Now after the Lucas' Colloquium I will also think of it in terms of [Lucas 1978] which, of course, occurs in the title of this paper. The other reference in the title of this paper is now fruitfully ambiguous. Originally, I intended it

to refer to my FME 2001 Berlin paper [Mac an Airchinnigh 2001] which gives a clear indication of how far the School has come. It is my first paper ever to the classified in the 'logic section' of a formal methods conference. I find that idea extremely amusing. Others may prefer to see as a self-reference to this paper which I am writing. I also find that idea quite amusing.

5.1 Acknowledgements

This paper has taken a very long time to produce and deliver. I wish to thank especially Dr. Bernhard Aichernig for encouraging me to get the paper written, for chasing me to get it finished and for being very patient over the very long delay.

Special thanks are due to all the participants of the Lucas' Colloquium who endured the quick introduction to the $M_C^{\clubsuit}$ topos logic and its use in modelling. Thanks also to those including Cliff Jones and Peter Lucas who corrected some of my memories and dates during and after my presentation (2001-05-18). I hope that there are not too many errors left in this written account.

The paper was typeset on my new Apple Macintosh iBook 2001 (acquired in June). It is a very beautiful white slimline computer with wireless connection to networks. I think of our $M_C^{\clubsuit}$ as being aesthetically comparable.

I used the Blue Sky Research Textures 2.1.4 and the LaTeX2ϵ format. Vince Darley's BibTex 1.1.8 was used for the references. Inline category theoretic arrow diagrams were produced using Xy-pic 3.6 of Kristoffer H. Rose and Ross R. Moore. Postscript diagrams were produced using Adobe Illustrator 6.

References

[**Bjørner and Jones 1978**] Bjørner, D. and C. B. Jones (Eds.) (1978). *The Vienna Development Method: The Meta-Language, Lecture Notes in Computer Science **61***. Berlin: Springer-Verlag.

[**Blikle 1987**] Blikle, A. (1987, May). Denotational engineering or from denotations to syntax. Technical Report 605, Institute of Computer Science, Polish Academy of Sciences, Warsaw, Poland. See also *VDM Europe 1987, Lecture Notes in Computer Science* **252**:151-209, and *Science of Computer Programming*, **12**(3):207-253, September 1989.

[**Blikle 1990**] Blikle, A. (1990, February). Why denotational? remarks on applied denotational semantics. Technical Report 679, Institute of Computer Science, Polish Academy of Sciences, Warsaw, Poland. See also *Fundamenta Informaticae* **28**(1-2): 55-85 (1996).

[**Dummett 2000**] Dummett, M. ([1977] 2000). *Elements of Intuitionism* (Second ed.). Number 39 in Oxford Logic Guides. Oxford: Clarendon Press. [ISBN 0 19 850524 8].

[**Eilenberg 1974**] Eilenberg, S. (1974). *Automata, Languages, and Machines, Volume A*. New York: Academic Press.

[**Hughes 2001**] Hughes, A. (2001). *Ph.D. Thesis: Elements of an Operator Calculus*. University of Dublin, Trinity College, Dublin, Ireland: Department of Computer Science.

[**Lawvere and Schanuel 1997**] Lawvere, F. and S. Schanuel (1997). *Conceptual Mathematics, A first introduction to categories*. Cambridge: Cambridge University Press. [ISBN 0-521-47817-0].
NOTE: An earlier version was published by the Buffalo Workshop Press, 1991, with an Italian translation, Franco Muzzio &c editore spa in 1994.

[**Lucas 1978**] Lucas, P. (1978). On the formalization of programming languages. In D. Bjørner and C. B. Jones (Eds.), *The Vienna Development Method: The Meta-Language*, pp. 1–23. Berlin: Lecture Notes in Computer Science, 61, Springer-Verlag. [ISBN 3 540 08766 4].

[**Lucas 1985**] Lucas, P. (1985). On the versatility of knowledge representations. In E. J. Neuhold and G. Chroust (Eds.), *Formal Models in Programming*, pp. 143–56. Amsterdam: North-Holland. [ISBN 0 444 87888 2].

[**Mac an Airchinnigh 1990**] Mac an Airchinnigh, M. (1990). *Ph.D. Thesis: Conceptual Models and Computing*. University of Dublin, Trinity College, Dublin, Ireland: Department of Computer Science.

[**Mac an Airchinnigh 1991**] Mac an Airchinnigh, M. (1991). Tutorial Lecture Notes on the Irish School of the *VDM*. In S. Prehn and W. J. Toetenel (Eds.), *VDM'91, Formal Software Development Methods Volume 2: Tutorials, Lecture Notes in Computer Science **552***, pp. 141–237. Berlin: Springer-Verlag.

[**Mac an Airchinnigh 2001**] Mac an Airchinnigh, M. (2001). Towards a Topos Theoretic Foundation for the Irish School of Constructive Mathematics ($M_C^{\clubsuit}$). In J. N. Oliveira and P. Zave (Eds.), *FME 2001: Formal Methods for Increasing Software Productivity*, pp. 396–418. Berlin: Lecture Notes in Computer Science, 2021, Springer-Verlag. [ISBN 3 540 41791 5].

[**Mac Lane and Moerdijk 1992**] Mac Lane, S. and I. Moerdijk (1992). *Sheaves in Geometry and Logic, A First Introduction to Topos Theory*. New York: Springer-Verlag. [ISBN 0-387-97710-4].

[**Wolfram 1996**] Wolfram Research, Inc. (1996). *Mathematica* (3.0 ed.). Champaign, Urbana: Wolfram Research, Inc. The Program.