Review of ‘Understanding Z’

Roger Bishop Jones

ICL Defence Systems

ABSTRACT

This document consists of a critical review of the book ‘Understanding Z’ by M.Spivey.

7 January 2016

Review of ‘Understanding Z’

Roger Bishop Jones

ICL Defence Systems

1. OVERVIEW

The primary purpose of this book is to give a formal semantics for the Z notation.

The first chapter introduces the language by means of a short example, discusses the reasons for providing a formal semantics and the problems arising from ‘meta-circular’ definitions. It compares Z with VDM and with algebraic specification techniques.

Chapter 2 identifies the semantic domains, and Chapter 3 provides the abstract syntax and the semantic mappings.

Chapter 4 offers an enhancement to the semantics to support generic definitions, discusses apparent instances of referential opacity, alternative treatments of partial functions, and the relationship between Z and Clear.

The final chapter discusses four further topics of interest. Proof rules for some of the operations of the schema calculus are provided and are shown to be sound by reference to the semantics. A method is shown for consistently introducing new types, illustrated by the natural numbers. The relationship between specifications and implementations is discussed, as is the need for non-determinism.

A summary of notation is provided, which appears to cover the variant of Z used as a metalanguage, and an index of definitions is conveniently condensed onto two facing pages.

2. ANALYSIS

The book does not claim to be an introduction either to Z or to techniques for formally defining the semantics of specification languages. It is therefore suitable primarily for readers having some prior acquaintance with both topics. Domain theory is not a prerequisite. The author does takes care to explain and illustrate his techniques before applying them to the problem.

By addressing a subset of the language the complexity of the formal semantics has been contained, and a reader not previously expert in formal semantics might therefore benefit from a careful reading of the book. Reader’s without knowledge of Z will however find some difficulty, since Z is itself the meta-language in which the semantics is defined.

The book does not claim to provide a standard semantics for the Z notation. The metalanguage and the object language used in the book are distinct variants of Z, neither of which fully corresponds with other recently available Z documentation.

2.1. Is a formal semantics the way to understand Z?

An understanding of the semantics of a language depends not merely upon successfully identifying what each construct in the language denotes, but also upon an appreciation of the properties enjoyed by these denotations. These properties are needed for reasoning about specifications. It might be argued, that the primary purpose of a formal semantics, is to enable the derivation or justification of proof rules. Derivation of proof rules is a substantial task, in default of which the merits of the semantics must remain *sub-judice*.

The author does provide some proof rules derived from his semantics in section 5.1, these are illustrative and cover schema conjunction, schema disjunction and schema projection. Very little discussion is provided of the proof theoretic consequences of the main body of the semantics.

2.2. Does the book define the semantics of Z?

The semantics of Z is defined using (a different variant of) Z itself, supplemented in parts by a Plotkin style ‘structured operational semantics’. The author makes no attempt to conceal the difficulties arising from this approach, putting rather more vigour into identifying the pitfalls than into explaining how they are avoided. Nevertheless his analysis of the problems is incomplete. He observes that a meta-circular definition of the semantics of a programming language will have a trivial least-fixed-point, and may have several significantly different non-trivial fixed-points. He does not attempt to identify or eliminate any ambiguities which arise in this way.

While clearly aware of some of the special considerations which apply to expressive specification languages, he does not acknowledge that these prevent the intended semantics from appearing among the fixed points. This can however be established from the use of the axiom of regularity in defining the *world of sets* which is the principal semantic (co-)domain.

The author is probably correct in claiming that his semantics could be rendered directly in first order set theory without the use of Z as a metalanguage. But in asking the reader to accept this he calls for an act of faith on the part of the reader, and begs a question which the formal semantics of Z might have been expected to settle more conclusively.

Nevertheless, the meta-circular semantics will most probably be more accessible to the intended readership than a semantics written directly in first order set theory.

Passing over the small number of trivial errors in the formal semantics there still remain a number of issues which justify discussion about whether the semantics defined is or should be the semantics of Z.

2.3. Discussion points

2.3.1. Conservative Extensions

The semantics given makes it clear that there is in Z no requirement that specifications should be conservative extensions of the basic language. This has the undesirable consequence that specifications cannot be shown consistent without resort to the metalanguage, and that giving structure to such a consistency proof is made more difficult. A semantics could have been given in which extensions are either conservative or content free, and within such a semantic framework proof rules could be established enabling consistency proofs to be conducted in a well structured way in the object language. The practical disadvantages of the semantics in this respect are substantial for any user who needs to show his specifications consistent, and for tool developers who wish to provide support for the construction of consistency proofs.

2.3.2. Schema Negation

The semantic domains chosen for schemas do not admit a rendering of the semantics of schema operations which faithfully reproduces previous informal accounts. The effect upon the semantics of schema negation (not noted by the author) seems most unsatisfactory and is illustrated by the following example.

The negation of the schema:

FUN[p]
$f: \mathbf{N} \nrightarrow \mathbf{N}$

P

where P is some desired property, on the basis of previous informal accounts would be:

FUN[p]
$f: \mathbf{N} \nrightarrow \mathbf{N}$

$\neg$ P

but according to the formal semantics it will be:

FUN[p]
$f: \mathbb{P}(\mathbf{N} \times \mathbf{N})$

$\neg$ (function f $\wedge$ P)

This is because the information contained in the signature of the schema over and above the type of the names (in this case the fact that ‘f’ must be a *functional* relation) is merged in the denotation with the information in the body of the schema (in this case P).

When the complement is taken all the non-functional relations emerge.

2.3.3. The Axiom of Choice

The author clearly has reservations about the axiom of choice and has taken steps to ensure that the axiom is not necessary.

(i) ‘μ’, previously described as the choice operator and given the full force of the classical choice function in draft rules of reasoning, has now been cut back to an operator of ‘definite description’.

(ii) global generic definitions are required to be definite rather than loose.

That the choice function has been discarded is probably a good thing. The requirement that generic definitions should not be loose seems to be unnecessary and undesirable. Unnecessary, because the semantics does not depend, contrary to the author’s claim, upon the axiom of choice. It is true that the consistency of some loose generic specifications can only be shown using the axiom of choice. But any loosening of a tight specification could be proven consistent without the axiom, and hence so could any implementable loose specification.

Even if the author were correct in claiming that the axiom of choice is necessary to give a semantics to loose global generic definitions, the cure seems worse than the illness. Disallowing looseness forces overspecification, and might make an implementation of a data type incompatible with the Z library for wholly trivial reasons. It also makes proofs more difficult by imposing further proof obligations. On the other hand, adverse consequences of admitting the axiom of choice are difficult to identify. Objections to it seem to be of a philosophical rather than practical nature.

2.3.4. Partial Functions

The treatment of partial functions falls between the sort of treatment necessary in first order set theory and those advocated for the VDM specification language.

In first order set theory every term must be assigned a value in the domain of quantification of each interpretation. A choice must be made in defining function application about what value should result when a function is applied to a value outside its domain. The primitive predicates (equality and membership) are total; classical first order logic is used without modification.

In the VDM literature, application of a function outside its domain of definition does not yield a value in the domain of quantification, predicates may be partial, and the logic is three valued.

In Spivey's semantics terms do not necessarily denote values in the domain of quantification, but the logic remains boolean. All predicates are ultimately formed from the equality and membership predicates, each of which will yield true or false even if its arguments are undefined. Undefinedness is therefore eliminated in the construction of formulae from terms.

In his semantics Spivey assigns the value false to the equality or membership predicates when either of their arguments is undefined. The motivation behind this choice is not explained, nor its consequences. The author clearly finds equality thus defined insufficient for his own purposes. He defines in the metalanguage a strong equality, '·', yielding true when applied to two undefined terms. This is used extensively in defining the semantics, but is not present in the object language.

This treatment of partial functions results in the need for side conditions on specialisation of universally quantified formulae. Before a specialisation can take place, the term to which specialisation is proposed must be shown defined.

The axiom of reflection continues to hold, e.g.:

$$\vdash \forall\ x{:}\mathbf{N} \bullet\ x = x$$

applying the everywhere-undefined partial function over natural numbers to a natural number will always yield an undefined term of type **N**:

$$\vdash (\varnothing{:}\ \mathbf{N} \nrightarrow \mathbf{N})\ 1 : \mathbf{N}$$

which by Spivey's semantics will not be equal to itself, hence:

$$\vdash \neg\ (\varnothing\ 1 = \varnothing\ 1)$$

Showing that specialisation of the law of reflection to the well typed term '∅ 1' cannot be permitted.

This is likely to make proofs more difficult than they would otherwise be.

3. CONCLUSIONS

This book sets a new standard in rigour of definition for the Z specification language. I hope that it does not prove to be the last word on the semantics of Z, but it does represent a very considerable improvement over previously available literature on this topic. The author has made a serious and careful attempt to put the semantics of Z on a sound basis, and students of the Z language will find this book a valuable aid in understanding Z.