Formal Aspects of Computing (1999) 11: 359–380

Formal Aspects of Computing

Revising Z: Part I – logic and semantics

Martin C. Henson[1] and Steve Reeves[2]

[1]Department of Computer Science, University of Essex, UK;
[2]Department of Computer Science, University of Waikato, New Zealand

Keywords: Specification language Z; Logics of specification languages; semantics of specification languages

Abstract. This is the first of two related papers. We introduce a simple specification logic Z_C comprising a logic and a semantics (in ZF set theory) within which the logic is sound. We then provide an interpretation for (a rational reconstruction of) the specification language Z within Z_C. As a result we obtain a sound logic for Z, including a basic schema calculus.

1. Introduction

1.1. Background

The specification language Z has been in existence, and has been very widely used, for more than a decade. There is, however, no definitive logical account of Z, although progress has been made (*e.g.* [WB92], [Bri95][1], [Nic95], [HM97], [Toy97], [BM96], [WD96], [Mar98]). These attempts to provide Z with a logic are not accompanied by meta-mathematical results such as a soundness proof; indeed, the logic in [Nic95] (and [BM96]) is inconsistent (see [Hen98] and note that [Mar98] repairs the error).

Our aim in this paper is to establish a logic for a version of Z, the development and nature of which has been guided by our associated mathematical investigations. This means that we have felt it necessary to allow the mathematical investigation to form, in part, a *critique* of the language Z as it is generally

Correspondence and offprint requests to: Martin Henson, Department of Computer Science, University of Essex, Wivenhoe Park, Colchester, Essex, C04 3SQ, UK

[1] This has now been superseded by a new thesis which is currently in the process of being examined. It contains a logic and is accompanied by a mathematical analysis. Brien's work is complementary to ours and will provide an opportunity for a fruitful comparative investigations in the future.

understood. The critique, and the revisionism which accompanies it, emerges most explicitly in the companion paper [HeR99]. Our own view is that developing a logic for Z long after establishing its *language* will inevitably suggest certain areas in which the language is best adjusted. Doubtless others may disagree, at least with the particular revisions we propose but, for our part, we have our mathematical analysis to resort to in substantiating claims of conceptual simplicity and elegance, and for defending our neologisms.

We do not offer this work as a final, nor even a definitive, account but we do feel that what follows establishes a methodologically sound trajectory for further work and fruitful comparison. Our work should, therefore, be considered as complementary to, rather than as a competitor of, the work which continues on the Z standard (which is, in part, working towards completing a logic for standard Z) and it will be very interesting to compare that with the approach we have taken here.

In discussing a logic for Z, it is also necessary to mention the related topic of semantics. Z semantics has, since [Spi88], been in a more well-developed state than the logic. This work has been substantially revised and updated by those responsible for the relevant sections of [Nic95]. It is quite possible that the semantic perspective has been given conceptual priority by those most concerned with these matters during the last decade. Although we would wish to argue strongly against such a priority, this will not be our major concern here, and it is only our surmise that that has been, in fact, the case. However, we would wish the reader to bear the following in mind in reading the paper: by giving the logic conceptual priority we are able to enormously *simplify* the semantics for Z. The most important consequence of this simplicity is that we will be able to provide a soundness proof for our logic (corollaries 5.3 and 5.5 below) with a very simple and straightforward proof. This observation should be contrasted with the situation in, for example, [Nic95] where there is no soundness proof, and the prospect that it would be enormously complicated to construct.

We have not, so far, mentioned proof-tools, but as [Mar97] has accurately diagnosed, the existence of a logic for Z should be considered a *precondition* for the proper development of proof-tools for Z, and is, therefore, a separate, though related, research area. It is, of course, a very important area and [Mar97] provides a comprehensive review of existing work to date, the most notable of which is perhaps the shallow semantic embedding of [KSW96].

1.2. Organisation of the paper

In Section 2 we introduce a specification system Z_C which is essentially a typed set theory incorporating the notion of a *schema type*. This system is, when compared with Z in its notational scope, very simple indeed. Z_C is much more than a notation: it is a specification *logic* and the language is associated with both rules for determining the types of terms, rules for determining that propositions are well-formed, and rules of inference. The meta-mathematical measure we investigate is the property of *syntactic consistency*: we show that all provably true (proto-)propositions are well-formed. Z_C forms an effective bridge between Z proper and the intended model in classical extensional set theory. In Section 3 we show how Z_C may be interpreted quite simply in ZF using, in particular, a suitable dependent product operation over a family of sets over a very small

(in ordinal terms) universe of sets. The interpretation is shown to be suitable by means of soundness results of a standard and, in this case, very simple kind.

In Section 4 we introduce a notation which is very much closer to the Z familiar in the literature. We include more substantial mechanisms for the construction of propositions, sets and, in particular, a basic algebra of schema operations. This notation is also equipped with a system of rules for type assignment for terms and propositionhood, together with a logic. As before, we can show that this system is syntactically consistent with respect to type assignment and propositionhood. In Section 5 we are able to describe an interpretation for this Z within the much simpler system Z_C. The interpretation has several significant features. We are able to test our interpretation by showing that it preserves the type assignment and propositionhood systems of both Z and Z_C. Finally we are able to demonstrate that the interpretation of Z in Z_C is sound: it preserves entailment. By composing results we complete the paper with a soundness theorem for Z in the intended model ZF.

The system we develop here is still a somewhat impoverished version of Z, and the companion paper [HeR99] contains extensions to the full system as we have developed it.

2. The specification logic Z_C

In this section we shall describe a simple specification logic which we call Z_C. It is based upon the notion of *schema type* which has been introduced in Z. Our strategy will be to interpret higher-level features of Z within this logic. The idea of interpreting the *language* of Z within a small core *language* is not new. The approach is novel in presenting a core specification *logic* and undertaking a systematic mathematical analysis.

2.1. The language of Z_C

We begin with types.

$$T \ ::= \ \mathbb{N} \mid \mathbb{P}\, T \mid T \times T \mid [D]$$

Schema types $[D]$ are explicitly part of the type system. We take $\mathbb{N}$ rather than $\mathbb{Z}$ to be the primitive type of Z_C. This is a more natural choice from a logical point of view and we may impose conventional rules for $\mathbb{N}$ from which other derived number systems, such as $\mathbb{Z}$, may be obtained.

Declarations are very simple.

$$D \ ::= \ l : T \mid l : T;\ D$$

Declarations of the form $l : T$ are called *prime* declarations. The labels l are *constants*. Prime declarations $l_0 : T_0$ and $l_1 : T_1$ are *compatible* unless $l_0 = l_1$ and $T_0 \neq T_1$. This idea extends naturally to schema types. We shall write $[D_0] \preceq [D_1]$ when the set of prime declarations of D_0 is a subset of that of D_1. Other meta-operations we shall need over (compatible) schema types: $[D_0] \setminus [D_1]$ is the schema type comprising all prime declarations of $[D_0]$ which do not occur in $[D_1]$. The schema type $[D_0] \vee [D_1]$ is the schema type comprising the union of the prime declarations in D_0 and D_1. As a matter of convention, which simplifies the

presentation, we take the ground instances of meta-syntactic expressions to be all and only those for which the relevant meta-operations are well-defined. This avoids laborious notation and extra sideconditions. For example writing $T_0 \vee T_1$ forces T_0 and T_1, in the context in which the expression appears, to be compatible schema types.

We also need the usual notion of substituting for a variable with a term. We denote the operation of substitution by $t_1[x/t_0]$ (in the meta-language) which means that all free occurrences of x are replaced by t_0 in t_1. This is extended in the usual way to allow simultaneous substitution of several variables.

Finally, we shall introduce meta-notational conventions which require substitution for labels. For this we need the *alphabet* operator, defined as follows: Let $[D] = [\cdots l_i : T_i \cdots]$. Then $\alpha[D] =_{df} \{\cdots l_i \cdots\}$ and we shall write $[\alpha[D]/t.\alpha[D]]$ to represent the family of substitutions: $\cdots \ [l_i/t.l_i] \ \cdots$

The formulae of Z_C delineate a typed predicate logic. We begin with the category which forms the basis for the language of well-typed formulae we require.

The *proto-syntax* of formulae is given by:

$$P \ ::= \ \bot \mid t = t \mid t \in t \mid \neg P \mid P \vee P \mid \exists z : T \bullet P$$

The logic of Z_C is classical and so the remaining logical connectives and the universal quantifier can be defined in terms of the above in the usual manner.

The proto-syntax of terms is as follows:

$$t \ ::= \ x \mid n \mid \{z : T \mid P\} \mid t.l \mid \langle\!| \cdots l \Rightarrow t \cdots |\!\rangle \mid t.1 \mid t.2 \\ \mid (t, t) \mid t \upharpoonright [D]$$

The last term formation operator is new and has no history in Z. We pronounce the symbol $\upharpoonright$ "filter" and its purpose is to permit the restriction of bindings to a given schema type. We shall see, in Section 5, how important these filtered terms are for constructing compositional interpretations for schema expressions. Our choice of notation indicates that there is an intimate relationship between filtered terms and restricted schemas; this is made clear in section 2.5 of [HeR99].

The subcategory of numerals is as expected:

$$n \ ::= \ 0 \mid succ \ n$$

For notational clarity we will use, in the sequel, the meta-variable C to range over set-terms. That is, terms which, under the well-formation system we now turn to, have the type $\mathbb{P} T$ for some T.

2.2. Type assignment and propositionhood in Z_C

Definition 2.1. Sequents of the system have the form: $\Gamma \rhd_C J$ where J is a judgement that a term has a type or that a proto-formula is a proposition. We will omit the subscript on the entailment symbol, whenever the context allows, in this system and in all the other systems which follow. Γ is a type assignment context for variables. Such contexts are understood to be partial functions from variables to types. They are extended as follows:

$$\Gamma, x : T =_{df} \lambda y. \begin{cases} T & x = y \\ \Gamma(y) & \text{otherwise} \end{cases}$$

For clarity of presentation we shall omit the entailment symbol and all components of contexts which are irrelevant to, or which remain unchanged by, any rule.

$$\frac{}{\bot\ prop}\ (C_\bot) \qquad \frac{t_0 : T \quad t_1 : T}{t_0 = t_1\ prop}\ (C_=) \qquad \frac{t : T \quad C : \mathbb{P}\, T}{t \in C\ prop}\ (C_\in)$$

$$\frac{P\ prop}{\neg P\ prop}\ (C_\neg) \qquad \frac{P_0\ prop \quad P_1\ prop}{P_0 \vee P_1\ prop}\ (C_\vee) \qquad \frac{x : T \vdash P\ prop}{\exists x : T \bullet P\ prop}\ (C_\exists)$$

$$\frac{}{x : T \vdash x : T}\ (C_x) \qquad \frac{}{0 : \mathbb{N}}\ (C_0) \qquad \frac{n : \mathbb{N}}{succ\ n : \mathbb{N}}\ (C_s)$$

$$\frac{x : T \vdash P\ prop}{\{x : T \mid P\} : \mathbb{P}\, T}\ (C_{\{\}}) \qquad \frac{t : [\cdots l_i : T_i \cdots]}{t.l_i : T_i}\ (C_.)$$

$$\frac{\cdots \quad t_i : T_i \quad \cdots}{\langle\!|\ \cdots l_i \Rrightarrow t_i \cdots\ |\!\rangle : [\cdots l_i : T_i \cdots]}\ (C_\Rrightarrow)$$

$$\frac{t : T_1 \times T_2}{t.1 : T_1}\ (C_1) \qquad \frac{t : T_1 \times T_2}{t.2 : T_2}\ (C_2)$$

$$\frac{t_0 : T_0 \quad t_1 : T_1}{(t_0, t_1) : T_0 \times T_1}\ (C_{()}) \qquad \frac{t : T_1 \quad T_0 \preceq T_1}{t \upharpoonright T_0 : T_0}\ (C_\upharpoonright)$$

Lemma 2.2. (The generation lemma for Z_C).

(i) If $\Gamma \vdash P_0 \vee P_1\ prop$ then $\Gamma \vdash P_0\ prop$ and $\Gamma \vdash P_1\ prop$
(ii) If $\Gamma \vdash \{x : T \mid P\} : T_0$ then $\Gamma, x : T \vdash P\ prop$ and $T_0 = \mathbb{P}\, T$

Proof. (*i*) There is only one rule with a conclusion of the form $\Gamma \vdash P_0 \vee P_1\ prop$, namely: $(C_\vee)$. (*ii*) Similarly.

These are two cases of a general result known as the *generation lemma*. There are many similar cases for all other conclusion forms and these are all proved in exactly the same manner.

2.3. Some consequences of typechecking

Before we move on to discuss the logic of Z_C there are number of auxiliary results which concern the rules for proposition formation and type assignment introduced in definition 2.1.

Lemma 2.3.
(i) If x does not appear free in P, and $x : T \vdash P\ prop$ then $P\ prop$
(ii) If x does not appear free in t, and $x : T_0 \vdash t : T_1$ then $t : T_1$
(iii) If $t : T_1$ and x does not appear free in t then $x : T_0 \vdash t : T_1$
(iv) If $P\ prop$ and x does not appear free in P then $x : T \vdash P\ prop$

Typing of terms is unique in Z_C.

Lemma 2.4. If $t : T_0$ and $t : T_1$ then $T_0 = T_1$

Proof. By the generation lemma (lemma 2.2). □

Lemma 2.5.

(*i*) If $P[z/t]$ *prop* and $t : T$ then $z : T \triangleright P$ *prop*

(*ii*) If $z : T \triangleright P$ *prop* and $t : T$ then $P[z/t]$ *prop*

(*iii*) If $t_1[z/t_0] : T_1$ and $t_0 : T_0$ then $z : T_0 \triangleright t_1 : T_1$

(*iv*) If $z : T_0 \triangleright t_1 : T_1$ and $t_0 : T_0$ then $t_1[z/t_0] : T_1$

Proof. The pairs (*i*) and (*iii*); (*ii*) and (*iv*) are each proved by simultaneous induction on the structure of the relevant derivations.

2.4. The logic of Z_C

The proto-judgements of the logic have the form:

$$\Gamma \vdash_C P$$

where a proto-context Γ has the form $\Gamma^-;\ \Gamma^+$. Γ^- is a type assignment context (a context for the type system) and Γ^+ is a set of formulae. These are well-formed according to the following rules.

$$\frac{}{\Gamma^-\ \mathit{context}} \qquad \frac{\Gamma^- \triangleright P\ \mathit{prop} \quad \Gamma\ \mathit{context}}{\Gamma^-;\ P, \Gamma^+\ \mathit{context}}$$

Proofs introduce new putative contexts, propositions and terms and the rules must be guarded in some cases by type judgements to ensure they remain type consistent. We shall establish that these conditions of well-formedness are maintained by the rules below (proposition 2.9). For notational simplicity we shall, in the sequel, often write t^T for a term t such that $t : T$ in the context in which it appears. For example, $\Gamma \vdash P(t^T)$ implies that, additionally, $\Gamma^- \triangleright t : T$.

Definition 2.6.

$$t_0 \equiv t_1 =_{df} \forall z \in t_0 \bullet z \in t_1 \wedge \forall z \in t_1 \bullet z \in t_0$$

Definition 2.7. (Logic of Z_C). As before, we omit all data which remain unchanged by a rule.

$$\frac{\Gamma \vdash P_0 \quad \Gamma^- \triangleright P_1\ \mathit{prop}}{\Gamma \vdash P_0 \vee P_1}\ (\vee_0^+) \qquad \frac{\Gamma \vdash P_1 \quad \Gamma^- \triangleright P_0\ \mathit{prop}}{\Gamma \vdash P_0 \vee P_1}\ (\vee_1^+)$$

$$\frac{P_0 \vee P_1 \quad P_0 \vdash P_2 \quad P_1 \vdash P_2}{P_2}\ (\vee^-)$$

$$\frac{\Gamma, P \vdash \bot \quad \Gamma^- \triangleright P\ \mathit{prop}}{\Gamma \vdash \neg P}\ (\neg^+) \qquad \frac{\neg\neg P}{P}\ (\neg^-) \qquad \frac{P \quad \neg P}{\bot}\ (\bot^+)$$

$$\frac{\Gamma \vdash \bot \quad \Gamma^- \triangleright P\ \mathit{prop}}{\Gamma \vdash P}\ (\bot^-)$$

$$\frac{\Gamma \vdash P[z/t] \quad \Gamma^- \triangleright t : T}{\Gamma \vdash \exists z : T \bullet P}\ (\exists^+)$$

$$\frac{\exists z : T \bullet P_0 \quad y : T;\ P_0[z/y] \vdash P_1}{P_1}\ (\exists^-), \text{for fresh } y$$

$$\frac{\Gamma, P \ context}{\Gamma, P \vdash P}\ (ass) \qquad \frac{\Gamma^- \rhd t : T}{\Gamma \vdash t = t}\ (ref)$$

$$\frac{\Gamma^- \rhd \langle\!| \cdots l_i \Rightarrow t_i \cdots |\!\rangle : T}{\Gamma \vdash \langle\!| \cdots l_i \Rightarrow t_i \cdots |\!\rangle.l_i = t_i}\ (\Rightarrow_0^=) \qquad \frac{\Gamma^- \rhd t : [\cdots l_i : T_i \cdots]}{\Gamma \vdash \langle\!| \cdots l_i \Rightarrow t.l_i \cdots |\!\rangle = t}\ (\Rightarrow_1^=)$$

$$\frac{\Gamma^- \rhd (t_1, t_2) : T}{\Gamma \vdash (t_1, t_2).1 = t_1}\ ((,)_0^=) \qquad \frac{\Gamma^- \rhd (t_1, t_2) : T}{\Gamma \vdash (t_1, t_2).2 = t_2}\ ((,)_1^=) \qquad \frac{\Gamma^- \vdash t : T_1 \times T_2}{\Gamma \vdash (t.1, t.2) = t}\ ((,)_2^=)$$

$$\frac{t_0 = t_1 \quad P[z/t_0]}{P[z/t_1]}\ (sub) \qquad \frac{P[n/0] \quad n : \mathbb{N};\ P \vdash P[n/succ\ n]}{n : \mathbb{N} \vdash P}\ (\mathbb{N}^-),\ \text{for fresh } n$$

$$\frac{\Gamma \vdash P[z/t] \quad \Gamma^- \rhd t : T}{\Gamma \vdash t \in \{z : T \mid P\}}\ (\{\}^+) \qquad \frac{t \in \{z : T \mid P\}}{P[z/t]}\ (\{\}^-)$$

$$\frac{t_0 \equiv t_1}{t_0 = t_1}\ (ext) \qquad \frac{\Gamma \vdash t.l_i = t_i \quad \Gamma^- \rhd t : T \quad [\cdots l_i : T_i \cdots] \preceq T}{\Gamma \vdash (t \upharpoonright [\cdots l_i : T_i \cdots]).l_i = t_i}\ (\upharpoonright^=)$$

Since contexts are sets we do not require structural rules. The following weakening rule is clearly admissible and may be usefully added:

$$\frac{\Gamma_0 \vdash P \quad \Gamma_0^-, \Gamma_1^-;\ \Gamma_0^+, \Gamma_1^+ \ context}{\Gamma_0^-, \Gamma_1^-;\ \Gamma_0^+, \Gamma_1^+ \vdash P}$$

2.5. Consequences of the logic for Z_C

There should also be a number of equality congruence rules for the terms of Z_C. These are not included in the system because they are all easily derivable; essentially because we have the rule (*sub*). The following are derived rules of Z_C:

$$\frac{t_1 = t_0}{t_0 = t_1}\ (sym) \qquad \frac{t_0 = t_1 \quad t_1 = t_2}{t_0 = t_2}\ (trans) \qquad \frac{t_0 = t_2 \quad t_1 = t_3}{(t_0, t_1) = (t_2, t_3)}\ (=_{(,)})$$

$$\frac{\Gamma \vdash t_0 = t_1 \quad \Gamma^- \rhd t_0 : T_1 \times T_2}{\Gamma \vdash t_0.1 = t_1.1}\ (=_{.1}) \qquad \frac{\Gamma \vdash t_0 = t_1 \quad \Gamma^- \rhd t_0 : T_1 \times T_2}{\Gamma \vdash t_0.2 = t_1.2}\ (=_{.2})$$

$$\frac{\cdots \quad t_{0i} = t_{1i} \quad \cdots}{\langle\!| \cdots l_i \Rightarrow t_{0i} \cdots |\!\rangle = \langle\!| \cdots l_i \Rightarrow t_{1i} \cdots |\!\rangle}\ (=_{\Rightarrow})$$

$$\frac{\Gamma \vdash t_0 = t_1 \quad \Gamma^- \rhd t_0 : [\cdots l_i : T_i \cdots]}{\Gamma \vdash t_0.l_i = t_1.l_i}\ (=_{.l})$$

$$\frac{\Gamma \vdash t_0 = t_1 \quad \Gamma^- \rhd t_0 : T_1 \quad T_0 \preceq T_1}{\Gamma \vdash t_0 \upharpoonright T_0 = t_1 \upharpoonright T_0}\ (=_{\upharpoonright})$$

We shall need the following definition, which extends filtering from terms to sets.

Definition 2.8. Let $T_0 \preceq T_1$. Let z and x be fresh variables.

$$C^{\mathbb{P} T_1} \upharpoonright T_0 =_{df} \{z : T_0 \mid \exists x : T_1 \bullet x \in C \wedge z = x \upharpoonright T_0\}$$

Then we have:

$$\frac{C : \mathbb{P} T_1 \quad T_0 \preceq T_1}{C \upharpoonright T_0 : \mathbb{P} T_0}\ (C_{\mathbb{P}\upharpoonright})$$

Then we have rules relating filtered terms and filtered sets:

$$\frac{\Gamma \vdash t \in C \quad \Gamma^- \rhd t : T_1 \quad T_0 \preceq T_1}{\Gamma \vdash t \upharpoonright T_0 \in C \upharpoonright T_0} \ (\in^+_\upharpoonright)$$

This follows by rules $(C_\upharpoonright)$, $(\{\}^+)$ and $(\exists^+)$.

$$\frac{\Gamma \vdash t \in C^{\mathbb{P} T_0} \upharpoonright T \quad \Gamma^-, x : T_0; \Gamma^+, x \in C, t = x \upharpoonright T \vdash P}{\Gamma \vdash P} \ (\in^-_\upharpoonright)$$

for fresh x. This follows by rules $(\{\}^-)$ and $(\exists^-)$.

2.6. Syntactic consistency for Z_C

The logic should only enable us to deduce well-formed propositions from well-formed assumptions. This is the content of the next result.

Proposition 2.9. (Syntactic consistency). If $\Gamma \vdash_C P$ when Γ *context* then $\Gamma^- \rhd_C P$ *prop*

Proof. By induction on the structure of the derivation $\Gamma \vdash P$. We give one case to illustrate. Suppose that Γ *context*.
Case rule $(\exists^-)$: we may assume *ex hypothesi* (first premise) that $\Gamma^- \rhd \exists z : T \bullet P_0$ *prop* since we have Γ *context* by assumption. From the generation lemma it follows that $\Gamma^-, z : T \rhd P_0$ *prop* and this, together with the assumption that Γ *context*, is sufficient to show that $\Gamma^-, z : T; \Gamma^+, P_0$ *context*. Since we know that the variable y is not free in P_0 we have, by alpha-conversion (in the meta-language), $\Gamma^-, y : T; \Gamma^+, P_0[z/y]$ *context* and then, *ex hypothesi* (second premise), that $\Gamma^-, y : T \rhd P_1$ *prop*. But since y is not free in P_1 this reduces to to $\Gamma^- \rhd P_1$ *prop* by lemma 2.3(*i*).

The specification logic Z_C is essentially a typed set theory in which, in particular, we have *schema* types. There are, however, no schemas in Z_C and this may seem rather odd since these are archetypical of Z. In fact, given the schema types, schemas are just special cases of the comprehensions. Specifically, we may introduce schemas by meta-notational convention using the following definition:

$$[\![D \mid P]\!] =_{df} \{z : [D] \mid P[\alpha[D]/z.\alpha[D]]\}$$

where z is a fresh variable. Note that this device requires us to allow the meta-variable P to range over the proto-propositions *extended with labels as terms*. The right-hand-side is, of course, (proto-)syntactically valid in Z_C. Given this definition we may provide the following versions of the comprehension rules using the schema notation:

$$\left[\frac{\Gamma \vdash P[\alpha[D]/t.\alpha[D]] \quad \Gamma^- \rhd t : [D]}{\Gamma \vdash t \in [\![D \mid P]\!]}\right]$$

$$\frac{t \in [\![D \mid P]\!]}{P[\alpha[D]/t.\alpha[D]]}$$

It is immediate that equality for schemas is also extensional (see definition 2.6). We shall return to this in far more detail in Section 5.2, where we show how an algebra of schemas can be represented in Z_C.

We are proposing that Z_C be taken as an adequate base theory within which the much higher-level features of Z can be interpreted. As such it plays an intermediate role between Z and classical, extensional set theory (which is the intended model for Z). To show that Z_C can play this role we must show that it can be faithfully interpreted in ZF, and we devote the next section to that task. Sections 4 and 5 are then devoted to showing that Z_C is adequate for the interpretation of Z.

3. A model of Z_C in ZF

In this section we provide an interpretation $[\![\_]\!]_C$ from the language of Z_C into ZF and present a variety of results. We shall omit the subscript on the interpretation function unless this is essential. The semantics is extremely simple. The novelty lies in our interpretation of schema types as dependent products over a family of sets from a small (in ordinal terms) cumulative universe.

3.1. Types

The language of types Z_C is given by a simple context free grammar. Such a grammar is understood mathematically to be an inductive definition over an operator which determines a set (the language of the grammar). The closure ordinal for this induction (the ordinal at which iteration of the operator reaches a fixpoint) is ω because the operator in question is continuous. In other words, the non-terminal operators may be applied finitely, but unboundedly, often. Consequently the type structures which can be described are, from a set-theoretic perspective, rather trivial. Consider, therefore, the following definition in ZF which constructs a tiny cumulative hierarchy.

$$
\begin{array}{lllll}
(i) & F(0) & = & \mathbb{N} \\
(ii) & F(\alpha+1) & = & F(\alpha) \cup \mathbb{P} F(\alpha) \\
(iii) & F(\omega) & = & \bigcup_{\alpha<\omega} F(\alpha)
\end{array}
$$

This function is guaranteed to exist by transfinite induction (in fact only transfinite induction below $\omega.2$ is required) and we then take $F(\omega+1)$ to be the universe within which the type system of Z_C may be interpreted. This universe is a *set*. Let B be an I-indexed family of sets over $F(\omega)$. (That is, $B \in I \to F(\omega+1)$.) Then we can define a *dependent function space* which is suitable for our purposes as follows:

$$\Pi_{(X \in I)}.B(X) =_{df} \{f \in I \to F(\omega) \mid (\forall i \in I)(f(i) \in B(i))\}$$

This we can harness to interpret the types of Z_C:

$$
\begin{array}{lllll}
(i) & [\![\mathbb{N}]\!] & =_{df} & \mathscr{N} \\
(ii) & [\![T_0 \times T_1]\!] & =_{df} & [\![T_0]\!] \mathscr{X} [\![T_1]\!] \\
(iii) & [\![\mathbb{P} T]\!] & =_{df} & \mathscr{P} [\![T]\!] \\
(iv) & [\![[\cdots l_i : T_i \cdots]]\!] & =_{df} & \Pi_{(X \in I)}.B(X)
\end{array}
$$

where $I =_{df} \{\cdots l_i \cdots\}$ and $B(l_i) =_{df} [\![T_i]\!]$. The labels l_i can be modelled in ZF in any number of ways, for example as finite ordinals. The only important point is that they be distinguishable from one another. We shall write them in ZF as we do in Z_C for simplicity.

3.2. Sets, logic and terms

We shall translate the entire proto-syntax of Z_C into ZF. We begin with the formulae.

$$
\begin{array}{lll}
[\![\bot]\!] & =_{df} & (\forall x)(\neg x = x) \\
[\![t_0 = t_1]\!] & =_{df} & [\![t_0]\!] = [\![t_1]\!] \\
[\![t_0 \in t_1]\!] & =_{df} & [\![t_0]\!] \in [\![t_1]\!] \\
[\![\neg P]\!] & =_{df} & \neg\, [\![P]\!] \\
[\![P_0 \vee P_1]\!] & =_{df} & [\![P_0]\!] \vee [\![P_1]\!] \\
[\![\exists\, z : T \bullet P]\!] & =_{df} & (\exists\, z)(z \in [\![T]\!] \wedge [\![P]\!])
\end{array}
$$

There are no special conditions to impose with respect to the judgement of propositionhood of Z_C, since ZF is an untyped language of sets. As a consequence we may interpret the judgement forms $\Gamma \triangleright_C P\ prop$ to mean that $[\![P]\!]$ is a well-formed formula of ZF. Since this is true for any P, the judgement is (*semantically*) redundant.

The terms are straightforwardly interpreted. We take the usual definition of cartesian product in ZF in which ordered pairs are defined by $\langle x, y\rangle =_{df} \{\{x\},\{x,y\}\}$. Then we make use of the maps $fst \in A\ \mathcal{X}\ B \to A$ such that $\langle a, b\rangle \overset{fst}{\mapsto} a$ and $snd \in A\ \mathcal{X}\ B \to B$ such that $\langle a, b\rangle \overset{snd}{\mapsto} b$.

In what follows let $f_0 \in [\![[\cdots l_i : T_i \cdots]]\!]$ and $f_0(l_i) = [\![t_i]\!]$ and $f_1 \in [\![[D]]\!]$ and $f_1(l) = [\![t]\!](l)$ when $l \in \alpha[D]$.

$$
\begin{array}{lll}
[\![x]\!] & =_{df} & x \\
[\![n]\!] & =_{df} & n \\
[\![\{x : T \mid P\}]\!] & =_{df} & \{[\![x]\!] \in [\![T]\!] \mid [\![P]\!]\} \\
[\![t.l]\!] & =_{df} & [\![t]\!](l) \\
[\![\langle\!| \cdots l_i \Rrightarrow t_i \cdots |\!\rangle]\!] & =_{df} & f_0 \\
[\![t.1]\!] & =_{df} & fst\ [\![t]\!] \\
[\![t.2]\!] & =_{df} & snd\ [\![t]\!] \\
[\![(t_0, t_1)]\!] & =_{df} & \{\{[\![t_0]\!]\},\{[\![t_0]\!], [\![t_1]\!]\}\} \\
[\![t \upharpoonright [D]]\!] & =_{df} & f_1
\end{array}
$$

3.3. Mathematical results

As a result of careful design the two crucial semantic results for Z_C are easy to prove.

Proposition 3.1. (Soundness of type assignment for Z_C). If $\Gamma \triangleright_C t : T$ then $[\![\Gamma]\!] \vdash_{ZF} [\![t]\!] \in [\![T]\!]$.

Proposition 3.2. (Soundness of Z_C logic). If $\Gamma \vdash_C P$ then $[\![\Gamma]\!] \vdash_{ZF} [\![P]\!]$.

4. Introducing Z

The specification logic Z which we introduce in this section will seem a somewhat impoverished version of the Z one routinely finds in the literature. Our intention is to provide a basic, high-level extension of Z_C which itself may be extended,

by further internal definition, in a variety of ways. It does not seem sensible to us that Z should aim to provide every feature for every conceivable application; particularly when these may be expressed very simply as notational conventions. What we focus on here will be generalisations in which sets may occur in what are type contexts in Z_C, and on the basic operations of the schema calculus.

There remain, nonetheless, a number of particularly important notions that it would be a mistake to leave unexamined so, having provided an interpretation for the specification logic of this section, we shall, in the companion paper [HeR99], go on to give an explanation of the most important derived constructs of Z.

4.1. The language of Z

We first give the proto-syntax for the language of Z which we consider in this paper. Essentially Z extends Z_C by allowing more general forms of propositions, more general forms of sets, and a number of new forms of terms. We shall use the same names for the syntactic categories as we used for Z_C, except for the declarations, since the Z_C category appears as well. In what follows we will always write D_C for the category of Z_C declarations, permitting us to reuse the category name D for the more general Z declarations.

Types are as they were in Z_C.

$$T ::= \mathbb{N} \mid \mathbb{P}\, T \mid T \times T \mid [D_C]$$

The proto-syntax for declarations in Z is, then:

$$D ::= l \in C \mid l \in C\,;\, D$$

The proto-syntax of propositions:

$$P ::= \bot \mid t = t \mid t \in C \mid \neg P \mid P \vee P \mid \exists z \in C \bullet P$$

Finally, we have the proto-syntax for terms. In addition to our use of C as a meta-variable for terms of type $\mathbb{P}\, T$ we now reserve the meta-variable S for sets of type $\mathbb{P}\, T$ where T is a schema type. Firstly, then, we have the terms which are sets:

$$t ::= \{z \in C \mid P\} \mid \mathbb{P}\, C \mid C \times C \mid \mathbb{N} \mid [D] \mid \lambda z \in C \bullet t$$

In Z, then, the types appear as a sub-category, or, more precisely, the *carrier sets* of the types do. These can be formally isolated by means of the following category definitions:

$$D^* ::= l \in T^* \mid l \in T^*;\, D^*$$
$$T^* ::= \mathbb{N} \mid T^* \times T^* \mid \mathbb{P}\, T^* \mid [D^*]$$

Since D^* is just the Z image of the Z_C declarations, we can take all the operations defined over D_C as inducing similar operations over D^*.

We have established a notational shift from conventional presentations of Z but, we feel, it is justified. Most particularly, allowing $t : C$ would suggest that we are permitting *sets to be types*, which we are not: such an approach would make typechecking undecidable. Writing $t \in T^*$ on the other hand suggests that we are permitting *types to be sets* which is precisely what Z allows. Additionally, the colon is a judgement of the type assignment and propositionhood system and this *never* assigns anything other than a type (*a name*) to a term. Consequently,

it would be somewhat confusing to permit more general usage for the type assignment symbol elsewhere in the language. These scruples arise here because we are dealing with logical systems and not simply the language. Perhaps we are being too fastidious, but the strict distinction between a type and its *carrier* is well recognised (*e.g.* [Spi92] p. 24). What might be a reasonable abuse of *language* (writing T in place of the carrier set T^*), we feel, may be more easily accommodated than an abuse of *logic* (writing $t : T$ in place of $t \in T^*$ (or $t \in T$)). Indeed we will not have to burden the presentation by distinguishing a type and its carrier precisely *because* we will insist on the correct logical relation in type judgements and in membership propositions.

Among the set comprehensions we shall, as before, isolate the *schemas* as a special case with the special meta-notation (where z is fresh):

$$[\![D \mid P]\!] =_{df} \{z \in [D] \mid P[\alpha[D]/z.\alpha[D]]\}$$

We then extend the term language with schema *expressions*:

$$t \ ::= \ \cdots \mid [\![D \mid P]\!] \mid S \vee S \mid S \setminus [l : T] \mid \neg S \mid S[l_1 \leftarrow l_0]$$

Note that all such expressions are included as sets in Z. We only use the unusual notation for renaming in order to prevent confusion with substitution in the meta-language. Schema hiding is, in standard approaches, equivalent to schema existential quantification (*e.g.* [WoD96] p. 181).

Disjunction, hiding, negation and renaming are sufficient to permit definitions for conjunction, implication, equivalence, pre-condition, composition and piping to be constructed using the usual definitions. We shall have more to say about this in the companion paper [HeR99].

The proto-syntax of terms is completed by means of the following:

$$t \ ::= \ \cdots \mid x \mid n \mid t.l \mid \langle\!| \cdots l \Rightarrow t \cdots |\!\rangle \mid t.1 \mid t.2 \mid (t, t) \mid t \upharpoonright [D] \mid \mathit{let}\ x == t\ \mathit{in}\ t \mid (\lambda z \in C \bullet t)t$$

We shall write $t_0 \mapsto t_1$ as a synonym for (t_0, t_1) when the pair is considered as a maplet, that is, as an element of a function.

4.2. Type assignment and propositionhood in Z

Definition 4.1. The judgements of the system again have the following forms:

$$\Gamma \rhd_Z P \ \mathit{prop}$$
$$\Gamma \rhd_Z t : T$$

The contexts, as usual, are partial functions giving type assignments for variables. As before, in giving the rules, we will omit any data which are not changed by a rule.

$$\frac{}{\bot \ \mathit{prop}}\ (Z_\bot) \qquad \frac{t_0 : T \quad t_1 : T}{t_0 = t_1 \ \mathit{prop}}\ (Z_=) \qquad \frac{t : T \quad C : \mathbb{P}\, T}{t \in C \ \mathit{prop}}\ (Z_\in)$$

$$\frac{P \ \mathit{prop}}{\neg P \ \mathit{prop}}\ (Z_\neg) \qquad \frac{P_0 \ \mathit{prop} \quad P_1 \ \mathit{prop}}{P_0 \vee P_1 \ \mathit{prop}}\ (Z_\vee) \qquad \frac{C : \mathbb{P}\, T \quad z : T \rhd P \ \mathit{prop}}{\exists z \in C \bullet P \ \mathit{prop}}\ (Z_\exists)$$

$$\frac{}{x : T \rhd x : T}\ (Z_x) \qquad \frac{}{0 : \mathbb{N}}\ (Z_0) \qquad \frac{n : \mathbb{N}}{\mathit{succ}\ n : \mathbb{N}}\ (Z_s)$$

$$\frac{S_0 : \mathbb{P}\, T_0 \quad S_1 : \mathbb{P}\, T_1}{S_0 \vee S_1 : \mathbb{P}(T_0 \vee T_1)}\ (Z_{\vee S}) \qquad \frac{S : \mathbb{P}\, T}{S \setminus [l : T_1] : \mathbb{P}(T \setminus [l : T_1])}\ (Z_h)$$

$$\frac{S : T}{S[l_0 \leftarrow l_1] : T[l_0 \leftarrow l_1]}\ (Z_r), l_1 \notin \alpha T$$

$$\frac{C : \mathbb{P}\, T \quad z : T \triangleright P\ prop}{\{z \in C \mid P\} : \mathbb{P}\, T}\ (Z_{\{\}}) \qquad \frac{S : T}{\neg S : T}\ (Z_{\neg S}) \qquad \frac{C : \mathbb{P}\, T}{\mathbb{P}\, C : \mathbb{P}\, \mathbb{P}\, T}\ (Z_{\mathbb{P}})$$

$$\frac{C_0 : \mathbb{P}\, T_0 \quad C_1 : \mathbb{P}\, T_1}{C_0 \times C_1 : \mathbb{P}(T_0 \times T_1)}\ (Z_\times) \qquad \frac{}{\mathbb{N} : \mathbb{P}\, \mathbb{N}}\ (Z_{\mathbb{N}})$$

$$\frac{\cdots \quad C : \mathbb{P}\, T \quad \cdots}{[\cdots l \in C \cdots] : \mathbb{P}[\cdots l : T \cdots]}\ (Z_{[]}) \qquad \frac{t : [\cdots l_i : T \cdots]}{t.l_i : T}\ (Z_.)$$

$$\frac{C : \mathbb{P}\, T_0 \quad z : T_0 \triangleright t : T_1}{\lambda z \in C \bullet t : \mathbb{P}(T_0 \times T_1)}\ (Z_\lambda) \qquad \frac{\lambda z \in C \bullet t_1 : \mathbb{P}(T_0 \times T_1) \quad t_0 : T_0}{(\lambda z \in C \bullet t_1)\ t_0 : T_1}\ (Z_{ap})$$

$$\frac{\cdots \quad t_i : T_i \quad \cdots}{\langle\!| \cdots l_i \Rrightarrow t_i \cdots |\!\rangle : [\cdots l_i : T_i \cdots]}\ (Z_\Rrightarrow)$$

$$\frac{t : T_1 \times T_2}{t.1 : T_1}\ (Z_1) \qquad \frac{t : T_1 \times T_2}{t.2 : T_2}\ (Z_2)$$

$$\frac{t_0 : T_0 \quad t_1 : T_1}{(t_0, t_1) : T_0 \times T_1}\ (Z_{()}) \qquad \frac{t_0 : T_0 \quad x : T_0 \triangleright t_1 : T_1}{let\ x == t_0\ in\ t_1 : T_1}\ (Z_{let})$$

$$\frac{t : [D_1] \quad [D_0] \preceq [D_1]}{t \upharpoonright [D_0] : [D_0]}\ (Z_\upharpoonright)$$

Similar results to those we exhibited for the corresponding system for Z_C hold for the system for Z.

Lemma 4.2. The generation lemma for Z holds.

Lemma 4.3.
(i) If x does not appear free in P, and $x : T \triangleright P\ prop$ then $P\ prop$
(ii) If x does not appear free in t, and $x : T_0 \triangleright t : T_1$ then $t : T_1$
(iii) If $t : T_1$ and x does not appear free in t then $x : T_0 \triangleright t : T_1$
(iv) If $P\ prop$ and x does not appear free in P then $x : T \triangleright P\ prop$

Typing of terms is also unique in Z.

Lemma 4.4. If $t : T_0$ and $t : T_1$ then $T_0 = T_1$

Proof. By the generation lemma (lemma 4.2).

Lemma 4.5.
(i) If $P[z/t]\ prop$ and $t : T$ then $z : T \triangleright P\ prop$
(ii) If $z : T \triangleright P\ prop$ and $t : T$ then $P[z/t]\ prop$
(iii) If $t_1[z/t_0] : T_1$ and $t_0 : T_0$ then $z : T_0 \triangleright t_1 : T_1$
(iv) If $z : T_0 \triangleright t_1 : T_1$ and $t_0 : T_0$ then $t_1[z/t_0] : T_1$

Lemma 4.6. $\triangleright\ T : \mathbb{P}\, T$ for all types T.

Proof. An easy induction on the structure of types. □

Lemma 4.7. The following rule is derivable:

$$\frac{C : \mathbb{P}\, T_1 \quad T_0 \preceq T_1}{C \upharpoonright T_0 : \mathbb{P}\, T_0}\ (Z_{\mathbb{P}\upharpoonright})$$

4.3. A logic for Z

Definition 4.8. (Logic of Z). The judgements of the logic have the form:

$$\Gamma \vdash_Z P$$

Contexts Γ have the form Γ^-; Γ^+ as they did in Section 2.4, and these are well-formed by means of analogous rules introduced earlier for Z_C.

$$\frac{\Gamma \vdash P_0 \quad \Gamma^- \rhd P_1\ prop}{\Gamma \vdash P_0 \vee P_1}\ (\vee_{\mathrm{o}}^+) \qquad \frac{\Gamma \vdash P_1 \quad \Gamma^- \rhd P_0\ prop}{\Gamma \vdash P_0 \vee P_1}\ (\vee_{\mathrm{I}}^+)$$

$$\frac{P_0 \vee P_1 \quad P_0 \vdash P_2 \quad P_1 \vdash P_2}{P_2}\ (\vee^-)$$

$$\frac{\Gamma, P \vdash \ \bot \quad \Gamma^- \rhd P\ prop}{\Gamma \vdash \neg P}\ (\neg^+) \qquad \frac{\neg\neg P}{P}\ (\neg^-) \qquad \frac{P \quad \neg P}{\bot}\ (\bot^+)$$

$$\frac{\Gamma \vdash \ \bot \quad \Gamma^- \rhd P\ prop}{\Gamma \vdash P}\ (\bot^-) \qquad \frac{P[z/t] \quad t \in C}{\exists z \in C \bullet P}\ (\exists^+)$$

$$\frac{\Gamma \vdash \exists z \in C \bullet P_0 \quad \Gamma^- \rhd C : \mathbb{P}\, T \quad \Gamma^-, y : T;\ \Gamma^+, y \in C, P_0[z/y] \vdash P_1}{P_1}\ (\exists^-)$$

for fresh y.

$$\frac{\Gamma, P\ context}{\Gamma, P \vdash P}\ (ass) \qquad \frac{\Gamma^- \rhd t : T}{\Gamma \vdash t = t}\ (ref)$$

$$\frac{t_0 = t_1 \quad P[z/t_0]}{P[z/t_1]}\ (sub)$$

$$\frac{\Gamma^- \rhd \langle\!| \cdots l_i \Rrightarrow t_i \cdots |\!\rangle : T}{\Gamma \vdash \langle\!| \cdots l_i \Rrightarrow t_i \cdots |\!\rangle.l_i = t_i}\ (\Rrightarrow^{=}_{\mathrm{o}}) \qquad \frac{\Gamma^- \rhd t : [\cdots l_i : T_i \cdots]}{\Gamma \vdash \langle\!| \cdots l_i \Rrightarrow t.l_i \cdots |\!\rangle = t}\ (\Rrightarrow^{=}_{\mathrm{I}})$$

$$\frac{\Gamma^- \rhd (t_1, t_2) : T}{\Gamma \vdash (t_1, t_2).1 = t_1}\ ((,)^{=}_{\mathrm{o}}) \qquad \frac{\Gamma^- \rhd (t_1, t_2) : T}{\Gamma \vdash (t_1, t_2).2 = t_2}\ ((,)^{=}_{\mathrm{I}}) \qquad \frac{\Gamma^- \vdash t : T_1 \times T_2}{\Gamma \vdash (t.1, t.2) = t}\ ((,)^{=}_{2})$$

$$\frac{P[z/t] \quad t \in C}{t \in \{z \in C \mid P\}}\ (\{\}^+) \qquad \frac{t \in \{z \in C \mid P\}}{t \in C}\ (\{\}^-_{\mathrm{o}}) \qquad \frac{t \in \{z \in C \mid P\}}{P[z/t]}\ (\{\}^-_{\mathrm{I}})$$

$$\frac{t \notin S}{t \in \neg S}\ (\neg S^+) \qquad \frac{t \in \neg S}{t \notin S}\ (\neg S^-)$$

$$\frac{t \in S}{t[l_0 \leftarrow l_1] \in S[l_0 \leftarrow l_1]}\ (S^+_{\leftarrow}), l_1 \notin \alpha S \qquad \frac{t \in S[l_0 \leftarrow l_1]}{t[l_1 \leftarrow l_0] \in S}\ (S^-_{\leftarrow}), l_1 \notin \alpha S$$

$$\frac{\Gamma \vdash t \in S \quad \Gamma^- \rhd t : T_0}{\Gamma \vdash t \upharpoonright T_0 \setminus [l : T_1] \in S \setminus [l : T_1]}\ (S^+_h)$$

$$\frac{\Gamma \vdash t^{\hat{T}} \in S^{\mathbb{P}\, T_0} \setminus [l : T_1] \quad \Gamma^-, y : T_0;\ \Gamma^+, y \in S, y \upharpoonright \hat{T} = t \vdash P}{\Gamma \vdash P}\ (S_h^-)$$

where $\hat{T}$ is $T_0 \setminus [l : T_1]$ and for fresh y.

$$\frac{\Gamma \vdash t \upharpoonright T_0 \in S_0 \quad \Gamma^- \rhd S_1 : \mathbb{P}\, T_1 \quad \Gamma^- \rhd t : T_0 \vee T_1}{\Gamma \vdash t \in S_0 \vee S_1}\ (S^+_{\vee_0})$$

$$\frac{\Gamma \vdash t \upharpoonright T_1 \in S_1 \quad \Gamma^- \rhd S_0 : \mathbb{P}\, T_0 \quad \Gamma^- \rhd t : T_0 \vee T_1}{\Gamma \vdash t \in S_0 \vee S_1}\ (S^+_{\vee_1})$$

$$\frac{t \in S_0^{\mathbb{P}\, T_0} \vee S_1^{\mathbb{P}\, T_1} \quad t \upharpoonright T_0 \in S_0 \vdash P \quad t \upharpoonright T_1 \in S_1 \vdash P}{P}\ (S^-_{\vee})$$

$$\frac{\Gamma^-, z : T;\ \Gamma^+, z \in C_0 \vdash z \in C_1 \quad \Gamma^- \rhd C_0 : \mathbb{P}\, T}{\Gamma \vdash C_0 \in \mathbb{P}\, C_1}\ (\mathbb{P}^+)$$

$$\frac{C_0 \in \mathbb{P}\, C_1 \quad t \in C_0}{t \in C_1}\ (\mathbb{P}^-)$$

$$\frac{t_1 \in C_1 \quad t_2 \in C_2}{(t_1, t_2) \in C_1 \times C_2}\ (\times^+) \qquad \frac{t \in C_1 \times C_2}{t.1 \in C_1}\ (\times^-_1) \qquad \frac{t \in C_1 \times C_2}{t.2 \in C_2}\ (\times^-_2)$$

$$\frac{}{0 \in \mathbb{N}}\ (\mathbb{N}^+_0)\ \frac{n \in \mathbb{N}}{succ\ n \in \mathbb{N}}\ (\mathbb{N}^+_1)\ \frac{P[n/0] \quad n : \mathbb{N};\ P \vdash P[n/succ\ n]}{n : \mathbb{N} \vdash P}\ (\mathbb{N}^-),\ \text{fresh } n$$

$$\frac{\cdots \quad t_i \in C_i \quad \cdots}{\langle\!|\, \cdots l_i \Rightarrow t_i \cdots \,|\!\rangle \in [\cdots l_i \in C_i \cdots]}\ ([]^+) \qquad \frac{t \in [\cdots l_i \in C_i \cdots]}{t.l_i \in C_i}\ ([]^-)$$

$$\frac{\Gamma \vdash t_0 \in C \quad \Gamma^- \rhd C : \mathbb{P}\, T_0 \quad \Gamma^-, z : T_0 \rhd t_1 : T_1}{\Gamma \vdash t_0 \mapsto t_1[z/t_0] \in (\lambda z \in C \bullet t_1)}\ (\lambda^+)$$

$$\frac{t_0 \in (\lambda z \in C \bullet t_1)}{t_0.1 \in C}\ (\lambda^-_0) \qquad \frac{t_0 \in (\lambda z \in C \bullet t_1)}{t_0.2 = t_1[z/t_0.1]}\ (\lambda^-_1)$$

$$\frac{\Gamma^- \rhd t_0 : T_0 \quad \Gamma^-, z : T_0 \rhd t_1 : T_1 \quad \Gamma \vdash t_0 \in C}{\Gamma \vdash (\lambda z \in C \bullet t_1)\ t_0 = t_1[z/t_0]}\ (\lambda^=)$$

$$\frac{\Gamma^- \rhd t_0 : T_0 \quad \Gamma^-, z : T_0 \rhd t_1 : T_1}{\Gamma \vdash let\ z == t_0\ in\ t_1 = t_1[z/t_0]}\ (let)$$

$$\frac{t_0 \equiv t_1}{t_0 = t_1}\ (ext) \qquad \frac{\Gamma \vdash t.l_i = t_i \quad \Gamma^- \rhd t : T \quad [\cdots l_i : T_i \cdots] \preceq T}{\Gamma \vdash (t \upharpoonright [\cdots l_i : T_i \cdots]).l_i = t_i}\ (\upharpoonright^=)$$

4.4. Consequences of the logic for Z

There are, as was the case with Z_C, a large number of congruence rules for equality which are all derivable using substitution.

$$\frac{t_1 = t_0}{t_0 = t_1}\ (sym) \qquad \frac{t_0 = t_1 \quad t_1 = t_2}{t_0 = t_2}\ (trans) \qquad \frac{t_0 = t_2 \quad t_1 = t_3}{(t_0, t_1) = (t_2, t_3)}\ (=_{()})$$

$$\frac{\Gamma \vdash t_0 = t_1 \quad \Gamma^- \triangleright t_0 : T_1 \times T_2}{\Gamma \vdash t_0.1 = t_1.1}\ (=_{.1}) \qquad \frac{\Gamma \vdash t_0 = t_1 \quad \Gamma^- \triangleright t_0 : T_1 \times T_2}{\Gamma \vdash t_0.2 = t_1.2}\ (=_{.2})$$

$$\frac{\cdots \quad t_{0i} = t_{1i} \quad \cdots}{\langle\!|\, \cdots l_i \Rrightarrow t_{0i} \cdots \,|\!\rangle = \langle\!|\, \cdots l_i \Rrightarrow t_{1i} \cdots \,|\!\rangle}\ (=_{\Rrightarrow})$$

$$\frac{\Gamma \vdash t_0 = t_1 \quad \Gamma^- \triangleright t_0 : [\cdots l_i : T_i \cdots]}{\Gamma \vdash t_0.l_i = t_1.l_i}\ (=_{.l})$$

$$\frac{\Gamma \vdash C_0 = C_1 \quad \Gamma^- \triangleright \{z \in C_0 \mid P\} : \mathbb{P}\, T}{\Gamma \vdash \{z \in C_0 \mid P\} = \{z \in C_1 \mid P\}}\ (=_{\{\}}) \qquad \frac{C_0 = C_1}{\mathbb{P}\, C_0 = \mathbb{P}\, C_1}\ (=_{\mathbb{P}})$$

$$\frac{C_0 = C_2 \quad C_1 = C_3}{C_0 \times C_1 = C_2 \times C_3}\ (=_{\times})$$

$$\frac{S_0 = S_2 \quad S_1 = S_3}{S_0 \vee S_1 = S_2 \vee S_3}\ (=_{\vee})$$

$$\frac{S_0 = S_1}{S_0 \setminus [l : T] = S_1 \setminus [l : T]}\ (=_{\setminus}) \qquad \frac{S_0 = S_1}{S_0[l_0 \leftarrow l_1] = S_1[l_0 \leftarrow l_1]}\ (=_{\leftarrow}), l_1 \notin \alpha S_0$$

$$\frac{S_0 = S_1}{\neg S_0 = \neg S_1}\ (=_{\neg})$$

$$\frac{\Gamma \vdash C_0 = C_1 \quad \Gamma, z : T \vdash t_0 = t_1 \quad \Gamma^- \triangleright C_0 : \mathbb{P}\, T}{\Gamma \vdash \lambda z \in C_0 \bullet t_0 = \lambda z \in C_1 \bullet t_1}\ (=_{\lambda})$$

for fresh z.

$$\frac{\Gamma \vdash t_0 = t_2 \quad \Gamma^- \triangleright t_0 : T \quad \Gamma, x : T \vdash t_1 = t_3}{\Gamma \vdash let\ x == t_0\ in\ t_1 = let\ x == t_2\ in\ t_3}\ (=_{let})$$

$$\frac{\lambda z \in C_0 \bullet t_0 = \lambda z \in C_1 \bullet t_1 \quad t_2 = t_3 \quad t_2 \in C_0}{(\lambda z \in C_0 \bullet t_0)\ t_2 = (\lambda z \in C_1 \bullet t_1)\ t_3}\ (=_{app})$$

$$\frac{\Gamma \vdash t_0 = t_1 \quad \Gamma^- \triangleright t_0 : T_1 \quad T_0 \preceq T_1}{\Gamma \vdash t_0 \upharpoonright T_0 = t_1 \upharpoonright T_0}\ (=_{\upharpoonright})$$

Set equality in Z is, like in Z_C, extensional. The necessary rules are also part of the logic for Z. The rule $(\in^+_{\upharpoonright})$, that relates filtered terms and filtered sets in Z_C also generalises to Z.

Lemma 4.9. The following rule is admissible:

$$\frac{t : T}{t \in T}$$

Proof. By induction on the structure of the term t. For example:
Case $t.1$: we may assume that $t.1 : T_1$ for some type T_1. By lemma 4.2 we have $t : T_1 \times T_2$ for some type T_2. *Ex hypothesi* we then obtain $t \in T_1 \times T_2$ and then $t.1 \in T_1$, by rule $(\times^-_1)$, as required.

Schemas were introduced in Z, as in Z_C, by convention. This induces immediately the following rules:

$$\frac{P[\alpha[D]/t.\alpha[D]] \quad t \in [D]}{t \in [D \mid P]}\ (S^+) \qquad \frac{t \in [D \mid P]}{P[\alpha[D]/t.\alpha[D]]}\ (S_{\text{o}}^-) \qquad \frac{t \in [D \mid P]}{t \in [D]}\ (S_{\text{I}}^-)$$

It is then possible to prove those relationships which are commonly used to describe (occasionally to define) the schema operators in the literature. This begins the task of establishing an equational logic for Z which is justified by the logic. The equations all have premises which ensure that the equalities are well-formed.

Lemma 4.10.

$$\frac{\Gamma^- \triangleright [D^* \mid P] : \mathbb{P}[D^*]}{\Gamma \vdash \neg[D^* \mid P] = [D^* \mid \neg P]}\ (\neg^=)$$

Proof. Note that the declarations must range over types. It is well known that this equation fails if the declarations range over sets in general. Assume that $\Gamma^- \triangleright [D^* \mid P] : \mathbb{P}[D^*]$. This implies that the equation is a proposition and that both sides have the type $\mathbb{P}[D^*]$.
Case ($\subseteq$): let $t : [D^*]$. Suppose that $t \in \neg[D^* \mid P]$. Using rule $(\neg S^-)$ this is $t \notin [D^* \mid P]$, or equivalently $\neg(t \in [D^*] \land P[\alpha[D^*]/t.\alpha[D^*]])$. Using De Morgan's law, this is just $t \notin [D^*] \lor \neg P[\alpha[D^*]/t.\alpha[D^*]]$. Assume that $t \notin [D^*]$. Using lemma 4.9 we obtain $t \in [D^*]$ from the assumption and hence we conclude that $\bot$, whence, by rule $(\bot^-)$, $\neg P[\alpha[D^*]/t.\alpha[D^*]]$. This now also follows by rule $(\lor^-)$ from the disjunction above. From this, and $t \in [D^*]$, we finally conclude, by rule (S^+), that $t \in [D^* \mid \neg P]$ as required.
Case ($\supseteq$): let $t : [D^*]$. Suppose that $t \in [D^* \mid \neg P]$. Using rules (S_{o}^-) and (S_{I}^-) we obtain: $\neg P[\alpha[D^*]/t.\alpha[D^*]]$ and $t \in [D^*]$. By rule $(\{\}^+)$ and propositional logic we then conclude that $t \notin [D^* \mid P]$ which, by rule $(\neg S^+)$, is $t \in \neg[D^* \mid P]$ as required. □

Lemma 4.11.

$$\frac{\Gamma^- \triangleright [D_0^* \mid P_0] : \mathbb{P}\, T_0 \quad \Gamma^- \triangleright [D_1^* \mid P_1] : \mathbb{P}\, T_1}{\Gamma \vdash [D_0^* \mid P_0] \lor [D_1^* \mid P_1] = [D_0^* \lor D_1^* \mid P_0 \lor P_1]}\ (\lor^=)$$

It is important to remember that in our framework labels and variables are distinct, consequently there is no possibility of global variable capture to be addressed here. However in the case of hiding, the distinction between variables and labels leads to a minor notational variation:

Lemma 4.12.

$$\frac{\Gamma^- \triangleright [D^* \mid P] : \mathbb{P}\, T_1}{\Gamma \vdash [D^* \mid P] \setminus [l : T] = [D^* \setminus [l : T] \mid \exists z : T \bullet P[l/z]]}\ (\setminus^=)$$

for fresh z.

In addition we have an equation relating general declarations over sets to declarations over types. This, by iteration, enables us to remove all non-type sets from the declarations of Z schemas in the equational logic.

Lemma 4.13.

$$\frac{\Gamma^- \rhd [D\,;\, l \in C \mid P] : \mathbb{P}\, T_1 \quad \Gamma^- \rhd C : \mathbb{P}\, T}{\Gamma \vdash [D\,;\, l \in C \mid P] = [D\,;\, l \in T \mid l \in C \wedge P]}\ (\in^{=})$$

4.5. Syntactic consistency for Z

We must, of course, ensure that the rules of the logic are syntactically consistent.

Proposition 4.14. If $\Gamma \vdash_Z P$ when Γ *context* then $\Gamma^- \rhd_Z P$ *prop*

Proof. By induction on the structure of the derivation $\Gamma \vdash P$. We give one case for illustration. Suppose that Γ *context*.
Case rule $(S_\vee^-)$: We have, immediately, that: $\Gamma^- \rhd S_0 : \mathbb{P}\, T_0$ and $\Gamma^- \rhd S_1 : \mathbb{P}\, T_1$. Using lemma 4.2 we have, *ex hypothesi* from the first premise, that: $\Gamma^- \rhd t : \mathbb{P}(T_0 \vee T_1)$. Then, because $T_0 \preceq T_0 \vee T_1$ and $T_1 \preceq T_0 \vee T_1$, it follows, by rule $(Z^\restriction)$, that $\Gamma^- \rhd t \restriction T_0 : T_0$ and $\Gamma^- \rhd t \restriction T_1 : T_1$. Hence we have, using rule $(Z_\in)$: $\Gamma^- \rhd t \restriction T_0 \in S_0$ *prop* and $\Gamma^- \rhd t \restriction T_1 \in S_1$ *prop*. This establishes that: $\Gamma, t \restriction T_0 \in S_0$ *context* and $\Gamma, t \restriction T_1 \in S_1$ *context*. Finally, using just the first of these, we conclude: $\Gamma^- \rhd P$ *prop*, *ex hypothesi* from the second premise of the rule.

5. An interpretation of Z in Z_C

In this section we describe a translation $[\![\_]\!]_Z$ of Z, as described above, into Z_C, our core specification logic. This translation (unlike normalisation processes for Z which are found in the literature) is compositional. This can be achieved because we have made precise the notions of propositionhood and type assignment. Indeed, since well-formed Z sentences are those which satisfy the rules of definition 4.1, we shall make use, where necessary, of the type information associated with Z terms. As before, we omit the subscript on the translation function unless it is essential.

5.1. Propositions

The language of Z propositions is only marginally more complicated than that of Z_C. The translation is, for the main part, transparent:

(i)	$[\![\bot]\!]$	$=_{df}$	$\bot$
(ii)	$[\![t_0 = t_1]\!]$	$=_{df}$	$[\![t_0]\!] = [\![t_1]\!]$
(iii)	$[\![t \in C]\!]$	$=_{df}$	$[\![t]\!] \in [\![C]\!]$
(iv)	$[\![\neg P]\!]$	$=_{df}$	$\neg\, [\![P]\!]$
(v)	$[\![P_0 \vee P_1]\!]$	$=_{df}$	$[\![P_0]\!] \vee [\![P_1]\!]$
(vi)	$[\![\exists z \in C^{\mathbb{P}T} \bullet P]\!]$	$=_{df}$	$\exists z : T \bullet z \in [\![C]\!] \wedge [\![P]\!]$

5.2. Schema terms

The three basic Z schema calculus operations can be translated into Z_C as follows. Note that we only interpret type-correct Z and, as a consequence, we may make use of the relevant typing information.

Let x and z be a fresh variables, then:

$$
\begin{array}{lll}
(i) & [\![\neg S^{\mathbb{P}\,T}]\!] & =_{df} \{z : T \mid z \notin [\![S]\!]\} \\
(ii) & [\![{S_0}^{\mathbb{P}\,T_0} \vee {S_1}^{\mathbb{P}\,T_1}]\!] & =_{df} \{z : T_0 \vee T_1 \mid z \upharpoonright T_0 \in [\![S_0]\!] \vee z \upharpoonright T_1 \in [\![S_1]\!]\} \\
(iii) & [\![S^{\mathbb{P}\,T_0} \setminus [l : T_1]]\!] & =_{df} \{z : T_0 \setminus [l : T_1] \mid \\
 & & \qquad \exists x : T_0 \bullet x \in [\![S]\!] \wedge z = x \upharpoonright T_0 \setminus [l : T_1]\}
\end{array}
$$

5.3. Set terms

There are a number of new forms of set available in Z, translated as follows. Again, z is a fresh variable:

$$
\begin{array}{lll}
(i) & [\![[D \mid P]^{\mathbb{P}\,T}]\!] & =_{df} \{z : T \mid z \in [\![[D]]\!] \wedge [\![P[\alpha T / z.\alpha T]]\!]\} \\
(ii) & [\![\{z \in C^{\mathbb{P}\,T} \mid P\}]\!] & =_{df} \{z : T \mid z \in [\![C]\!] \wedge [\![P]\!]\} \\
(iii) & [\![\mathbb{P}\, C^{\mathbb{P}\,T}]\!] & =_{df} \{z : \mathbb{P}\,T \mid \forall x : T \bullet x \in z \Rightarrow x \in [\![C]\!]\} \\
(iv) & [\![{C_1}^{\mathbb{P}\,T_1} \times {C_2}^{\mathbb{P}\,T_2}]\!] & =_{df} \{z : T_1 \times T_2 \mid z.1 \in [\![C_1]\!] \wedge z.2 \in [\![C_2]\!]\} \\
(v) & [\![\mathbb{N}]\!] & =_{df} \{z : \mathbb{N} \mid z = z\} \\
(vi) & \left[\!\!\left[[\cdots l_i \in C_i^{\,T_i} \cdots]\right]\!\!\right] & =_{df} \{z : [\cdots l_i : T_i \cdots] \mid \\
 & & \qquad \cdots \wedge z.l_i \in [\![C_i]\!] \ \wedge \cdots\} \\
(vii) & [\![\lambda x \in C^{\mathbb{P}\,T_1} \bullet t^{\,T_2}]\!] & =_{df} \{z : T_1 \times T_2 \mid \\
 & & \qquad z.1 \in [\![C]\!] \wedge z.2 = [\![t[x/z.1]]\!]\}
\end{array}
$$

5.4. Other terms

In addition to the sets, which are translated above, there are two new forms of term in Z. In full the translation is:

$$
\begin{array}{lll}
(i) & [\![x]\!] & =_{df} x \\
(ii) & [\![n]\!] & =_{df} n \\
(iii) & [\![t.l]\!] & =_{df} [\![t]\!].l \\
(iv) & [\![\langle\!| \cdots l_i \Rrightarrow t_i \ \cdots |\!\rangle]\!] & =_{df} \langle\!| \cdots l_i \Rrightarrow [\![t_i]\!] \ \cdots |\!\rangle \\
(v) & [\![t.1]\!] & =_{df} [\![t]\!].1 \\
(vi) & [\![t.2]\!] & =_{df} [\![t]\!].2 \\
(vii) & [\![(t_0, t_1)]\!] & =_{df} ([\![t_0]\!], [\![t_1]\!]) \\
(viii) & [\![let\ x == t_0\ in\ t_1]\!] & =_{df} [\![t_1]\!][x / [\![t_0]\!]] \\
(ix) & [\![(\lambda z \in C \bullet t_1)\ t_0]\!] & =_{df} [\![t_1]\!][z / [\![t_0]\!]] \\
(x) & [\![t \upharpoonright T]\!] & =_{df} [\![t]\!] \upharpoonright T
\end{array}
$$

Lemma 5.1. $[\![P[x/t]]\!] = [\![P]\!][x / [\![t]\!]]$

Proof. Variables are unchanged by the translation and this is sufficient.

We will use this property without further reference in the sequel.

5.5. Correctness of the translation

The syntax of the systems Z and Z_C are both given in two parts: a proto-syntax equipped with rules for type assignment and propositionhood. Our translation supposes that the Z expressions it considers are well-formed, but yields expressions which are *prima facie* only in the proto-syntax of Z_C. It is incumbent upon us to show that the translation preserves syntactic well-formedness. This is the content of the following proposition.

Proposition 5.2.
(i) If $\Gamma \rhd_Z P \ prop$ then $\Gamma \rhd_C [\![P]\!] \ prop$
(ii) If $\Gamma \rhd_Z t : T$ then $\Gamma \rhd_C [\![t]\!] : T$

Proof. By simultaneous induction on the structure of the antecedent derivations. We give one case for illustration.
Case rule $(Z_\exists)$: we have $\Gamma \rhd_C [\![C]\!] : \mathbb{P} T$ *ex hypothesi* from which, by lemma 2.3(*iii*), we may conclude that $\Gamma, z : T \rhd_C [\![C]\!] : \mathbb{P} T$ for fresh z. From this and the instance of axiom (C_x) $\Gamma, z : T \rhd_C z : T$ we have $\Gamma, z : T \rhd_C z \in [\![C]\!] \ prop$ by rule $(C_\in)$. From this and $\Gamma, z : T \rhd_C [\![P]\!] \ prop$, which follows *ex hypothesi*, we may conclude that $\Gamma, z : T \rhd_C z \in [\![C]\!] \wedge [\![P]\!] \ prop$ by (derived) rule $(C_\wedge)$. Then by rule $(C_\exists)$ we have $\Gamma \rhd_C \exists z : T \bullet z \in [\![C]\!] \wedge [\![P]\!] \ prop$ which is $\Gamma \rhd_C [\![\exists z \in C \bullet P]\!] \ prop$ as required.

As a corollary we have the soundness of the type assignment system in ZF. We temporarily write $[\![\_]\!]$ for the composition $[\![\_]\!]_C \circ [\![\_]\!]_Z$.

Corollary 5.3. If $\Gamma \rhd_Z t : T$ then $[\![\Gamma]\!] \vdash_{ZF} [\![t]\!] \in [\![T]\!]$

Proof. Compose propositions 3.1 and 5.2.

Next we have the relative soundness result for the logic.

Proposition 5.4. If $\Gamma^-; \Gamma^+ \vdash_Z P$ then $\Gamma^-; [\![\Gamma^+]\!] \vdash_C [\![P]\!]$

Proof. By induction on the structure of the antecedent derivation. We provide one case for illustration.
Case $(\exists^+)$:
Ex hypothesi we have $[\![P]\!][z / [\![t]\!]]$ and $[\![t]\!] \in [\![C]\!]$. By (derived) rule $(\wedge^+)$ we have $(z \in [\![C]\!] \wedge [\![P]\!])[z / [\![t]\!]]$ and, using proposition 2.9, $[\![t]\!] : T$ for some T. Hence, by rule $(\exists^+)$, $\exists z : T \bullet z \in [\![C]\!] \wedge [\![P]\!]$ which is $[\![\exists z \in C^{\mathbb{P} T} \bullet P]\!]$ as required. □

Finally, as a corollary, we have soundness for the logic in ZF.

Corollary 5.5. If $\Gamma \vdash_Z P$ then $[\![\Gamma]\!] \vdash_{ZF} [\![P]\!]$

Proof. Combine propositions 5.4 and 3.2.

This together with corollary 5.3 completes the process of modelling Z in ZF.

6. Conclusions

We have introduced a specification logic Z_C which is sound with respect to a simple set-theoretic semantics. We have gone on to demonstrate that the basic apparatus of the specification language Z can be defined within Z_C. As a result the extended logic is both conservative over Z_C, and, as a corollary of this,

consistent. Much of the technical development concerns the interplay between the type systems and the logic proper: we have spelled out what it means for terms to be well-typed and, correspondingly, what it means for a proposition to be well-formed. Inference is shown to preserve these properties, and we have established that the translation of the more elaborate system into Z_C preserves both well-formation and inference properties. Our logics contrast with the current draft of the Z standard (Annex F of [Nic95]) in several respects. That system should, but does not, preserve well-formation of terms and propositions, there exists no soundness proof with respect to the independently given semantics and there is no clear technical connection which links the logic of well-formation with the logic proper. In these regards, in particular, our formulation is an improvement.

We have, however, not gone far enough. The extended logic described here is still impoverished in several important respects, notably the absence of a calculus for schemas. The standard logic [Nic95] is also incomplete in this area. It is the purpose of the companion paper "Revising Z: Part II - logical development" [HeR99] to make good these omissions and to continue the critique we have begun here.

Acknowledgements

We would like to thank the Department of Computer Science at the University of Waikato, New Zealand, the Centre for Discrete Mathematics and Theoretical Computer Science, New Zealand, the Royal Society of Great Britain, and the EPSRC (grant number GR/L57913) for financial assistance which has supported the development of this research. We are most grateful to Doug Goldson, Lindsay Groves, Ian Toyn, Ray Turner and Mark Utting for many useful discussions. We are particularly grateful to the Editor-in-Chief, Cliff Jones, for his help and encouragement in the preparation of the final version of this paper. Our final thanks go to one of the referees, who performed the most thorough and fruitful task of refereeing that we have ever experienced.

References

[BrM96] Brien, S. and Martin, A.: A tutorial of proof in standard Z. Technical report, Technical monograph PRG-120, University of Oxford, 1996.

[Bri95] Brien, S.: A model and logic for generically typed set theory (Z). Technical report, (draft) D. Phil. thesis, University of Oxford, 1995.

[Hen98] Henson, M. C.: The standard logic for Z is inconsistent. *Formal Aspects of Computing Journal*, 10(3): 243–247, 1998.

[HaM97] Hall, J. and Martin, A.: *W* reconstructed. In *Proceedings ZUM '97, LNCS 1212*, pages 115–134. Springer, 1997.

[HeR99] Henson, M. C. and Reeves, S.: Revising Z: Part II - logical development. *Formal Aspects of Computing Journal*, 11(4): 381–401, 1999.

[KSW96] Kolyang, Santen, B. and Wolff, B.: A structure preserving encoding of *Z* in Isabelle/HOL. In *Proceedings Formal Methods Europe*. LNCS Vol. 1051, Springer Verlag, 1996.

[Mar97] Martin, A.: Approaches to proof in Z. Technical report, Technical Report TR97-34, SVRC, University of Queensland, 1997.

[Mar98] Martin, A.: A revised deductive system for Z. Technical report, Technical Report TR98-21, SVRC, University of Queensland, 1998.

[Nic95] Nicholls, J. (ed.): *Z Notation: Version 1.2*. Z Standards Panel, 1995.

[Spi88] Spivey, J. M.: *Understanding Z: A specification language and its formal semantics*. C.U.P., 1988.
[Spi92] Spivey, J. M.: *The Z notation: A reference manual*. Prentice Hall, 1992.
[Toy97] Toyn, I. (ed.): *Z Notation: Draft 0.8*. Unpublished draft, 1997.
[WoB92] Woodcock, J. and Brien, S.: W: A logic for Z. In *Proceedings of ZUM '91, 6th Conf. on Z*. Springer Verlag, 1992.
[WoD96] Woodcock, J. and Davies, J.: *Using Z: Specification, Refinement and Proof*. Prentice Hall, 1996.

Note added in proof

Since this paper was accepted for publication, the draft Z standard [Nic95] has been superseded by the Final Committee Draft Standard [Toy99]. Our comments regarding semantics remain germane. On the other hand, the Z standards panel has decided to remove the logic entirely, rather than to correct it. As a result, our paper now becomes, at least in part, a work of advocacy in favour of a logic for Z.

[Toy99] Toyn, I. (ed.): Z Notation: Final Committee Draft, CD 13568.2, ftp://ftp.york.ac.uk./hise_reorts/cadiz/ZSTAN/fcd.ps.gz, 1999

Received March 1998
Accepted in revised form April 1999 by C. B. Jones