Problems with Z

Roger Bishop Jones
ICL Defence Systems
1. Z REFERENCE DOCUMENTATION

A What is the best reference material currently available covering each of the following areas:

1 Concrete Syntax
2 Type System
3 Semantics
4 Proof Theory
5 Methodology

B What further developments to such reference material are anticipated?

C Is there any way we can be sure of being kept up to date on such material?

D Is there in progress or in prospect any standardisation activity for Z?

E If Z is still fluid, can we exert any influence on the direction of development, and if so, how?
2. THE SYNTAX OF Z

A Why are there two different signs for introducing definitions (\(\hat{=}, :=\))?

B When is infix/prefix usage permitted?

C What is the current story on boxes?

3. THE Z TYPE-SYSTEM

A Under what conditions are recursive type definitions legal?

B Is \(\mathbb{N}\) a built in type or not? (also seq)

C We infer that the things after colon’s are arbitrary sets, is this correct?

D Are "dependent" schema types permitted?

E Any hope of "type inheritance" for schema types?

F What are the rules for type parameterisation?
4. THE SEMANTICS OF Z

A Disjoint Union

B Recursive types

C N and seq

D The use of schema types for specifying operations.

E The semantic significance of decorations.

5. PROOF THEORY

A What is it?

B How do we ensure or demonstrate consistency of specifications?

6. METHODOLOGY

A What is it?
7. SPECIFYING OPERATIONS AS FUNCTIONS

Type of Object
AUTO

Type of Property
P AUTO

Type of Operation
\[
\text{IN} \times \text{STATE} \rightarrow \text{STATE} \times \text{OUT} \\
\subseteq \mathcal{P}(\text{IN} \times \text{STATE} \times \text{STATE} \times \text{OUT})
\]

Type of Property of Operation
\[
\mathcal{P} (\text{IN} \times \text{STATE} \rightarrow \text{STATE} \times \text{OUT})
\]

Type of Non-Deterministic Operation
\[
\text{IN} \times \text{STATE} \rightarrow \mathcal{P} (\text{STATE} \times \text{OUT})
\]

Type of Property of Non-Deterministic Operation
\[
\mathcal{P} (\text{IN} \times \text{STATE} \rightarrow \mathcal{P} (\text{STATE} \times \text{OUT}))
\]

Type of Partial Operation
\[
\text{IN} \times \text{STATE} \rightarrow \text{STATE} \times \text{OUT}
\]

Type of Property of Partial Operation
\[
\mathcal{P} (\text{IN} \times \text{STATE} \rightarrow \text{STATE} \times \text{OUT})
\]

Type of Partial Non-Deterministic Operation
\[
\text{IN} \times \text{STATE} \rightarrow \mathcal{P} (\text{STATE} \times \text{OUT})
\]

Type of Property of Partial Non-Deterministic Operation
\[
\mathcal{P} (\text{IN} \times \text{STATE} \rightarrow \mathcal{P} (\text{STATE} \times \text{OUT}))
\]
8. SPECIFYING OPERATIONS USING SCHEMA TYPES

\[ \Delta \text{STATE} \equiv [i:\text{IN}; s,s':\text{STATE}; o:\text{OUT}] \]
\[ \equiv \text{IN} \times \text{STATE} \times \text{STATE} \times \text{OUT} \]

Not necessarily an operation at all. Certainly not a loose specification of an operation.

Can we find any interpretations of schema types as specifications of operations?

First define the type of functional subtypes of the schema type \( \Delta \text{STATE} \).

\[ \text{PAUTO} \equiv \{x: \mathbb{P} \Delta \text{STATE} | \]
\[ \forall y, z: \Delta \text{STATE}. (y \in x \land z \in x \land y.i = z.i \land y.s = z.s) \Rightarrow y.o = z.o \land y.s' = z.s' \} \]

We could then interpret a schema type as a lose specification denoting all of its (partial) functional subtypes.

Alternatively we might only wish to admit total functional subtypes:

\[ \text{AUTO} \equiv \{x: \text{PAUTO} | \forall i: \text{IN}, s: \text{STATE}. \]
\[ \exists s': \text{STATE}, o: \text{OUT}, y: \Delta \text{STATE}. \]
\[ y.i = i \land y.s = s \land y.s' = s' \land y.o = o \} \]
A further interpretation is as a single non-deterministic operation: