Haroon Rashid

Reg#: 16549

Semester:6th

Sessional Assignment: Software Verification and Validation

Submitted To: Sir ZAIN SHAUKAT

Question:

What is Z specification, why it is use for, also give Example.

Answer:

Definition

Z specification: Z is a model oriented formal specification language based on Zermelo-Fränkel axiomatic set theory and first order predicate logic. It is a mathematical specification language, with the help of which natural language requirements can be converted into mathematical form.

Introduction: With the ever-increasing complexity of computer systems, reliable and effective, design and development of high-quality systems that satisfy their requirements is extremely important. In the mission and safety critical system failure can cause cost overrun, loss of lives or even severe economic consequences can arise. So, in such situations, it is necessary that errors are uncovered before software is put into operation. These challenges call for acceptance of proper engineering methods and tools and have motivated the use of formal methods in software engineering. There are varieties of formal specification languages available to fulfill this goal and one way to achieve this goal is by using Z formal specification language. Z is model oriented formal method based on set theory and first order predicate calculus.

Description of Z Formal Specification Language: The Z language is a model oriented, formal specification language that was proposed by Jean-Raymond Abrail, Steve Schuman and Betrand Meyer in 1977 and it was later further developed at the programming research group at Oxford University. It is based on Zermelo Fränkel axiomatic set theory and first order predicate logic. The Z notation is a strongly typed, mathematical, specification language. It has robust commercially available tool support for checking Z texts for syntax and type errors in much the same way that a compiler checks code in an executable programming language. It cannot be executed, interpreted or compiled into a running program. It allows specification to be decomposed into small pieces called schemas. The schema is the main feature that distinguishes Z from other formal notations. In Z, both static and dynamic aspects of a system can be described using schemas. The Z specification describes the data model, system state and operations of the system. Z specification is useful for those who find the requirements, those who implement programs to meet those requirements, those who test the consequences, and those who write instruction manuals for the system.

Z also helps in refinement towards an implementation by mathematically relating the abstract and concrete states. Z is being used by a wide variety of companies for many different applications.

In the Z notation there are two languages: Mathematical Language The mathematical language is used to describe various aspects of a design: objects and the relationships between them by using propositional logic, predicate logic, sets, relation and functions.

Schema Language The schema language is used to structure and compose descriptions: collecting pieces of information, encapsulating them, and naming them for reuse.

Why it uses for:

- A Z specification forces the software developer to completely analyze the problem domain. (e.g. identify the state space and pre and post conditions for all operations).
- A Z specification forces all major design decisions to be made prior to coding the implementation. Coding should not commence until you are certain about what you should be coding.
- A Z specification is a valuable tool for generating test data, and the conformance testing of completed systems.
- A Z specification allows formal exploration of properties of system.
- The flexibility to model a specification which can directly lead to the code.
- A large class of structural models can be described in Z without higher – order features, and can thus be analyzed efficiently.
- Independent Conditions can be easily added later.

Example in Z

Step1:

We look at the following C function:

```
int f(int a) { int i, term, sum;

term = 1; sum = 1; for (i = 0; sum <= a; i++)
 {term = term + 2; sum = sum + term;}
 return i;
}
```

What does this code do?

Step2:

Again, but with documentation:

```
int iroot(int a) // Integer square root { int i, term, sum;
term = 1; sum = 1; for (i = 0; sum <= a; i++) {
term = term + 2; sum = sum + term;
 }
 return i;
}
```

How does this code work?

Step3:

The name and the comment for iroot are not as helpful as they might seem.

• Some numbers don’t have integer square roots.. What happens if you call iroot with a set to 3?

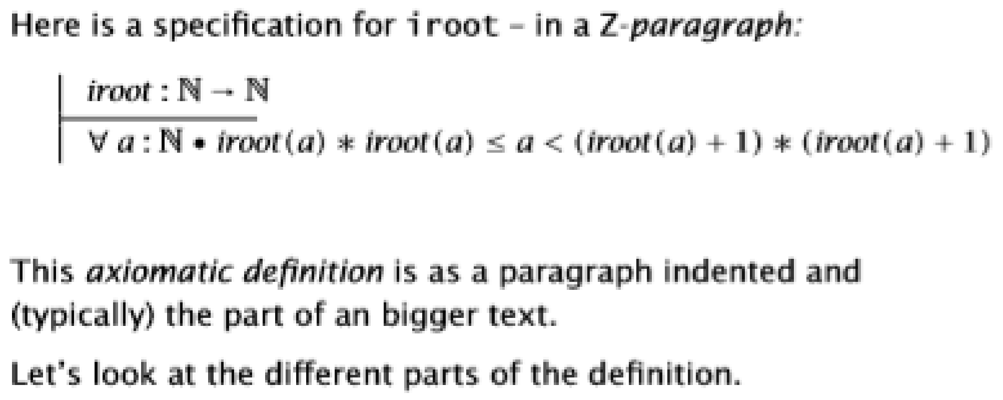
Step4:

The name and the comment for iroot are not as helpful as they might seem.
• Some numbers don’t have integer square roots. What happens if you call iroot with a set to 3?
• For negative numbers integer square roots are not defined (in R). What happens if you call iroot with a set to -4?

Step5:

The name and the comment for iroot are not as helpful as they might seem.
• Some numbers don’t have integer square roots. What happens if you call iroot with a set to 3?
• For negative numbers integer square roots are not defined (in R). What happens if you call iroot with a set to -4?
Conclusion: Names and comments are not enough to describe the behavior completely.

Step6:



Here is a specification for `iroot` – in a Z-*paragraph:*

$$\begin{array}{|l} iroot : \mathbb{N} \rightarrow \mathbb{N} \\ \hline \forall a : \mathbb{N} \bullet iroot(a) * iroot(a) \leq a < (iroot(a) + 1) * (iroot(a) + 1) \end{array}$$

This *axiomatic definition* is as a paragraph indented and (typically) the part of an bigger text.

Let’s look at the different parts of the definition.

Step7:

The declaration of iroot

iroot : N→N corresponds to the C-declaration
int iroot(int a)

Recognize: iroot doesn’t receive negative numbers and also returns none.

Step8:

The Predicate

$\forall a : \mathbb{N} \bullet iroot(a) * iroot(a) \leq a < (iroot(a)+1) * (iroot(a)+1)$ shows that iroot returns the biggest integer square root:

iroot(3)=1

iroot(4)=2

iroot(8)=2

iroot(9)=3

The predicate corresponds to the C function definition – it describes only what the function does without explaining how to do it.

Conclusion:

- Z is one of the numbers of specification languages which are being developed around the world. Z can be used to compactly specify real systems (ATM). Z has various collection of library (Mathematical Toolkit), which supports user to specify the requirements without any ambiguity.
- Large specifications are achievable in Z, using the schema notation for structuring. Also it is possible to produce hierarchical specifications. A part of a system is specified in isolation, and then put into a global context.
- By applying formal method in terms of Z notation, it is observed that it does not require a high level of mathematics rather it requires knowledge of basic set theory and first order logic for the analysis of a complete system.
- Difficulties with Z are cannot do concurrency, Timing aspects, Algorithmic aspects and programming constraints, and Sequencing operations.