Formal Method in Software Engineering (SE-313)

Course Teacher
Assistant Professor
Engr. Mustafa Latif

Introduction to Z Formal Specification

Z Formal Specification

Sets

- **Sets were discussed in previous slides in general, these slides on their use in Z.**
- **The Z notation is based upon set theory; specifications in Z find their meanings as operations upon sets.**

3

Z Formal Specification

Set

- **Definition: A set is an unordered collection of distinct objects.**
- **We can build sets containing any objects that we like, but usually we consider sets whose elements have some property in common.**
- **The objects comprising a set are called its elements or members.**
- **Note that:**
  - **– sets do not contain duplicates;**
  - **– the order of elements in a set is not significant.**

4

Z Formal Specification

Set

- By convention, a set can be defined by enumerating its components in curly brackets.
- EXAMPLE.
  - Vowels == {a,e,i,o,u}
  - Weekend == {Sat; Sun}
- The double-equals symbol (==) is read 'is defined to be'; it gives us a method for naming sets, so that we can subsequently use them.
- (We can't use sets unless they have been previously defined in some way.)

5

Z Formal Specification

Set

- In Z, another equivalent method for defining sets is allowed . . .
- EXAMPLE.

$$Vowels ::= a \mid e \mid i \mid o \mid u$$

- Another standard convention is to allow number ranges . . .
- EXAMPLE.
  - Days == 1.. 365

6

Z Formal Specification

Set

- **Some sets occur so often that they have been pre-defined and given standard names.**
  - **$\mathbb{N}$ the natural numbers: $\mathbb{N}$ = {0,1,2,3,…}**
  - **$\mathbb{N}_1$ the natural numbers greater than 0:**
    - **$\mathbb{N}$ = {1,2,3,…}**
- **$\mathbb{Z}$ the integers: $\mathbb{Z}$ = {…,-2,-1,0,1,2,…}**
- **$\mathbb{B}$ the truth values: $\mathbb{B}$ = {true, false}**

7

Z Formal Specification

Relational Operators for Sets

- **The set theory in Z provides us with a variety of relational operators for expressing the properties of sets.**

Set Membership

- **To express the fact that an object x is a member of set S, we write x ∈ S**
- **Note that an expression with this form is either true or false; the thing is either a member or not.**
- **Examples.**
  - **a∈ $\{a, e, i, o, u\}$; 1 ∈ $\mathbb{N}$; 23 ∈ Days**

8

Z Formal Specification

Relational Operators for Sets

- Note that the r.h.s. of the expression x ∈ S must be an expression which returns a set of values of the same type as the expression on the l.h.s returns.
  - EXAMPLE. This isn't allowed: {1,2,3} ∈ {4,5,6}
  - But this is: {1,2,3} ∈ {{1,2,3}, {6,7,8}} and {1,2,3} ∈ {{6,2,1}}
- The first legal expression above evaluates to true; the second evaluates to false.
- Note that Z differs in this respect from what might be called naive set theory: Z is said to be a typed set theory, (because we must check the types on the l.h.s and r.h.s. of our expressions.)

9

Z Formal Specification

Relational Operators for Sets

Set Equality

- Definition: Let S and T be arbitrary sets of the same type. Then S = T is true iff S and T contain precisely the same members.
  - Note that the order in which values occur in the set is not significant.
  - EXAMPLES. The following is a true
  - expression: {1,2,3} ={2,3,1} and Days = 1…365 and Days = ℕ

10

Z Formal Specification

Relational Operators for Sets

Subsets

- **Definition: Let S and T be arbitrary sets of the same type. Then S ⊆ T is true iff every member of S is also a member of T.**
  - **EXAMPLES. The following are true expressions:**

$$\{1, 2, 3\} \subseteq \mathbb{N};$$
$$\{true\} \subseteq \mathbb{B};$$
$$\mathbb{N} \subseteq \mathbb{Z};$$
$$\mathbb{N}_1 \subseteq \mathbb{N};$$
$$\{1, 2, 3\} \subseteq \{1, 2, 3\}.$$

whereas the following are not:

$$\{1, 2, 3, 4\} \subseteq \{1, 2, 3\};$$
$$\mathbb{Z} \subseteq \mathbb{N}.$$

-

11

Z Formal Specification

Relational Operators for Sets

- **Note that equal sets are subsets of each other; we can express this fact in the following theorem about sets:**
  - **(S = T)) ⟹(S ⊆ T)**

Proper Subsets

- **Definition: Let S and T be arbitrary sets of the same type. Then: S ⊂ T is true iff both S ⊆ T and S ≠ T.**
- **If S ⊂ T then then S is said to be a proper subset of T.**

The Empty Set

- **There is a special set that has the property of having no members; this set is called the empty set, and is denoted either {} or ∅**
- **• Note that if x is any value, then the expression x ∈ ∅; must be false. (Nothing is a member of the empty set.)**
- **• Similarly, if S is an arbitrary set, then ; ∅ ⊆ S must be true; this includes of course ; ∅ ⊆ ∅ ; (The emptyset is a subset of every set.)**

12

Z Formal Specification

Relational Operators for Sets

The Powerset Operator

- **Definition: Let S be an arbitrary set. Then the set of all subsets of S is given by $\mathbb{P}$ S:**
  - EXAMPLES.
    - If
      $S = \{1\}$
      then
      $\mathbb{P} S = \{\emptyset, \{1\}\}$
    - If
      $S = \{1, 2\}$
      then
      $\mathbb{P} S = \{\emptyset, \{1\}, \{2\}, \{1, 2\}\}$
    - If
      $S = \{a, b, c\}$
      then
      $\mathbb{P} S = ?$

a

13

Z Formal Specification

Relational Operators for Sets

The Powerset Operator

- **We can again use , to introduce $\mathbb{P}$ as a derived operator:**
  - **T $\in \mathbb{P}$ S $\Longrightarrow$ (T $\subseteq$ S)**
  - **Note that if S has n members, then $\mathbb{P}$ S has 2^n members.**
- **Example 5.14 Four friends have been invited to dinner: Ali, Bilal, Cashif, and Danish. If their names are abbreviated to A, B, C, and D, then the set of people that actually arrive will be an element of the power set**

$$\begin{aligned}\mathbb{P}\{A, B, C, D\} = \{ &\emptyset, \{A\}, \{B\}, \{C\}, \{D\}, \{A, B\}, \{A, C\}, \{A, D\}, \\ &\{B, C\}, \{B, D\}, \{C, D\}, \{A, B, C\}, \{A, B, D\}, \\ &\{A, C, D\}, \{B, C, D\}, \{A, B, C, D\} \}\end{aligned}$$

a

14

Z Formal Specification

Manipulating Sets

Set Union

- The union of two sets is a third set that contains the members of both.
- Definition: Let S and T be arbitrary sets. Then x ∈ (S ∪ T) is true iff either x ∈ S or x ∈ T.
- • Note that we can introduce this as a derived operator:
  - x ∈ S T ⇔ x ∈ S ∨ x ∈ T
- EXAMPLES.

$$\{a, e, i\} \cup \{o, u\} = \{a, e, i, o, u\};$$
$$\{a, e\} \cup \{e, i\} = \{a, e, i\};$$
$$\emptyset \cup \{a, e\} = \{a, e\}.$$

15

Z Formal Specification

Manipulating Sets

Set Intersection

- The intersection of two sets is a third set that contains only elements common to both.
- Definition: Let S and T be arbitrary sets. Then x ∈ (S ∩ T)
- iff both x ∈ S and x ∈ T.
- As a derived operator: x ∈ (S ∩ T) ⇔ x ∈ S ∧ x ∈ T.
- EXAMPLES.

$$\{a, e, i\} \cap \{o, u\} = \emptyset;$$
$$a\neg \in \{a, e, i\} \cap \{e, i\};$$
$$\mathbb{N} = \mathbb{N} \cap \mathbb{Z}.$$

16

Z Formal Specification

Manipulating Sets

Set Difference

- **The notation S \ T denotes the set obtained from S by removing from it all the elements of T that occur in it; in other words, you take T away from S.**
- **Definition: Let S and T be arbitrary sets. Then x ∈ (S \ T) is true iff x ∈ S and x ∉ T.**

As a derived operator: x ∈ (S \ T) ⇔ x ∈ S ∧ x∉ T.

- EXAMPLES.

$$\{a, e, i, o, u\} \setminus \{o\} = \{a, e, i, u\}$$
$$\emptyset \setminus \{1, 2, 3\} = \emptyset.$$

17

Z Formal Specification

Manipulating Sets

Cardinality

- **The cardinality of a set is the number of elements in it.**
- **Definition: If S is an arbitrary set, then the cardinality of S is denoted #S.**
- **Note that for an arbitrary set S: #S ∈ ℕ**
- EXAMPLES.

$$\#\{1, 2, 3\} = 3;$$
$$\#\emptyset = 0;$$
$$\#(\{1, 2, 3\} \cap \{2\}) = 1.$$

18

Z Formal Specification

Cardinality

- **Example In the game of Cluedo™, it is assumed that a murder has been committed. The players are then invited to guess the identity of the person responsible, the room in which the crime was committed, and the weapon used. If we define the set of guests, the set of locations, and the set of potential weapons,**

$$Guests == \{ Mrs\ Peacock, Miss\ Scarlett, Reverend\ Green, Mrs\ White, Colonel\ Mustard, Professor\ Plum \}$$

$$Rooms == \{ Library, Study, Lounge, Hall, Kitchen, Billiard\ Room, Ballroom, Conservatory, Dining\ Room \}$$

$$Weapons == \{ Rope, Dagger, Revolver, Candlestick, Lead\ Pipe, Spanner \}$$

then the set of possible solutions is given by the Cartesian product:
(Gurst x Rooms x Weapons)
and a typical guess would be (Colonel Mustard; Library; Revolver)
It was Colonel Mustard, in the library, with the revolver.
If there are 6 guests, 9 rooms, and 6 weapons in Cluedo™ then there are 6 × 9 × 6 = 324 elements in the set guess.

19

Z Formal Specification

Defining Sets: Comprehension

- **We have seen how sets may be defined by numeration — listing their contents.**
- **This technique is impractical for large sets!**
- **Comprehension is a way of defining sets:**
  - **in terms of their properties;**
  - **in terms of other sets.**
- **Comprehension is best explained via**
- **examples:**

$$\{n : \mathbb{N} \mid (n \geq 0) \land (n \leq 3)\} = \{0, 1, 2, 3\}$$

$$\{n : \mathbb{Z} \mid (n > 1) \land (n < 1)\} = \emptyset$$

$$\{n : \mathbb{N} \mid (n \geq 0) \land (n \leq 3) \bullet n + 1\} = \{1, 2, 3, 4\}$$

20

Z Formal Specification

Defining Sets: Comprehension

- In general, a set comprehension expression will have three parts:
  - – a signature — specifies the base sets from which values are extracted;
  - – a predicate — the defining property of the set;
  - –a term — specifies how the actual values in the set are computed.
- Generally, we omit the term part — we just have a signature and a predicate, as in examples (1) and (2), above.

21

Z Formal Specification

Defining Sets: Comprehension

- Example Suppose that a red car is seen driving away from the scene of a crime. In this case, the authorities might wish to talk to anyone who owns such a vehicle. If Person denotes the set of all people, then the set to consider is given by
  - {x:Person | x drives a red car}
- A simple comprehension term {x : s | p} has two parts: a declaration part x : s and a predicate part p. The declaration part may be seen as a generator, providing a range s of possible values for x; the predicate part may be seen as a filter, picking out only those values of x that satisfy p.
- It may be that we are interested in some expression formed from the values satisfying the predicate, and not in the values themselves. In this case, we add a term part to our set comprehension: we write { x : s | p • e }

22

Z Formal Specification

Defining Sets: Comprehension

- **Example 5.9 In order to pursue their investigation of the crime, the authorities require a set of addresses to visit. This set is given by**
- **{ x: Person | x drives a red car • address(x)}**

23

Z Formal Specification

Types

- **In Z we can only form sets from objects that are similar in some way.**
- **Same set must have the same type; sets in Z are typed.**
- **{2,4, red, yellow, 6} [TYPE ERROR! Elements have different types.]**
- **Every type is a set, but not all sets are types.**
- **The natural numbers N are a set, not a type.**
- **Natural numbers belong to the type integer, Z, because every natural number is also an integer.**
- **An object's type is the most inclusive or maximal set to which it belongs.**

24

Z Formal Specification

Types

- Example A computer system used by the United States Immigration Service might store information about foreign nationals presently in the United States. In a specification of this system, the set of all people would be a good choice for a basic type. The set of all UK nationals would be a poor choice, as we are likely to consider supersets of this set.

25

Z Formal Specification

Types

- Additional types can be created using the power set constructor $\mathbb{P}$ and the Cartesian product $\times$.
- If T is a type, then the power set $\mathbb{P}$T is the type of all subsets of T .
- If T and U are types, then T × U is the type of all pairs formed from elements of T and elements of U .

26

Z Formal Specification

Types

- Example The power set $\mathbb{PZ}$ is the type of all sets of integers, $\{1, 2, 3\} \in \mathbb{PZ}$
- while the Cartesian product $\mathbb{Z} \times \mathbb{Z}$ is the type of all number pairs: $(1, 2) \in \mathbb{Z} \times \mathbb{Z}$

27

Z Formal Specification

Declarations

- The simplest way to define an object is to declare it.
- If the object is a type, then
  - [Type] introduces a new basic type called Type.
- If the object is a variable, then
  - x : A introduces a new variable x, drawn from the set A.
- A declaration of the form x : t, where t is a type, is called a signature.

28

Z Formal Specification

Declarations

- **Example A hotel switchboard uses a software package to maintain a record of call charges to current guests. A formal specification of this system could include the declaration**

  [Guest, Room]

  introducing two basic types to represent the set of all guests and the set of all rooms. A variable of the type Guest is introduced by the following declaration:

- **x : Guest**

29

Z Formal Specification

Constraining variables, axiomatic definitions

- **A second form of definition includes a constraint upon the object being introduced.**
- **Such definitions are said to be axiomatic, as the constraint is assumed to hold whenever the symbol is used: it is an axiom for the object.**
  - **where the predicate expresses the constraints upon the object.**

declaration

predicate

30

Z Formal Specification

Constraining variables, axiomatic definitions

- **The definition**

$$x : s \mid p$$

introduces a new symbol x, an element of s, satisfying predicate p.

- **Example We may use an axiomatic definition to define the set of natural numbers:**

$$\mathbb{N} : \mathbb{P}\mathbb{Z} \mid \forall z : \mathbb{Z} \bullet z \in \mathbb{N} \Leftrightarrow z \geq 0$$

31

Z Formal Specification

Constraining variables, axiomatic definitions

- **Example We may define a constant maxsize as follows:**

$$maxsize : \mathbb{N} \mid maxsize > 0$$

- **Variables defined in axiomatic definitions are global: They can be used anywhere in a Z text after their definition. This is called declaration before use.**

32

Z Formal Specification

Constants, abbreviation definitions

- A variable that can only take on one value is called a constant.
- Example: This axiomatic definition makes size a constant; it describes a single configuration, the one with two megabytes of memory:

$$size : \mathbb{N} \mid size = 2048$$

- The Z abbreviation definition is a shortcut for defining global constants.
- It uses the definition symbol == to introduce the variable and give its value on one line.
  - size = = 2048

33

Z Formal Specification

Constants, abbreviation definitions

- Example The abbreviation definition
  - Additive == {red, green, blue}
  - introduces a set Additive, as another name for the set described in enumeration above.
  - The names red, green, and blue must be defined elsewhere, they are not introduced by the abbreviation. If they are declared as elements of a type Colours, then Additive is a constant of type $\mathbb{P}$ Colours.
- Example Given the basic type Person, representing the set of all people, we may introduce abbreviations for the set of all people who take sugar in tea:
  - SweetPeople == {p : Person | p drinks tea ∧ p takes sugar}

34

Z Formal Specification

Normalization and signatures

- **A definition where the types are explicitly spelled out in this way is said to be normalized.**
- **A signature is a declaration that names the type (normalized definition).**
- **e : EVEN is a declaration, but e : Z is a signature.**

35

Z Formal Specification

Normalization and signatures

$$e : EVEN$$
$$o : ODD$$
$$p : PRIME$$

unnormalized

$$e, o, p : \mathbb{Z}$$
$$e \in EVEN$$
$$o \in ODD$$
$$p \in PRIME$$

normalized

- **The predicate $e \in EVEN$ expresses the same constraint as the declaration $e : EVEN$.**

36

Z Formal Specification

Set variables

- **Z variables can name whole collections of objects; that is, they can denote sets.**
- **In Z, variables whose values are sets are declared as follows, using the symbol ℙ (usually pronounced "set of").**
- **This axiomatic definition declares a set of natural numbers named PRIME and a set of numbers in the range 1 .. 6 named toss:**

$$\begin{array}{|l} PRIME : \mathbb{P}\ \mathbb{N} \\ toss\ : \mathbb{P}\ DICE \end{array}$$

- **Every set has a type. The type of a set whose elements have type T is ℙ T. So the type of PRIME and toss is both ℙ Z.**

37

Z Formal Specification

Defining new types

- **Z provides two kinds of paragraphs for introducing new types.**
- **Free types are similar to the enumerated types provided by many programming languages.**
- **To define a free type, give its name and then, after the definition symbol ::=, list all of its elements.**
- **For example, here is the declaration for the free type COLOR:**
  - **COLOR ::= red | green | blue | yellow | cyan | magenta | white | black**
  - **The order here is not significant — no sequence is implied.**

38

Z Formal Specification

Defining new types

- The second method is the basic type definition.
- We define a basic type when we do not want to say in advance what all the elements are; this is usual when sets are large.
- To define a basic type, simply mention its name, enclosed in brackets e.g [NAME] is the declaration for the type whose elements are all the names that might appear in a telephone directory.
- [TYPE1, TYPE2] declare more than one basic type in one paragraph
- Sets introduced both ways can be used later to declare variables. Here palette is a set of COLOR, and subscribers is a set of NAME.
-

$$palette : \mathbb{P}\ COLOR$$
$$subscribers : \mathbb{P}\ NAME$$

39

Z Formal Specification

Identifiers and layout

- Z is case sensitive, so name, Name, and NAME are three distinct identifiers..
- You can use semicolons instead of line breaks to separate declarations and predicates.

$$size: \mathbb{N}$$
$$level: \mathbb{N}$$
$$\hline$$
$$size = 2048$$
$$level = 2$$

$$size: \mathbb{N}; level: \mathbb{N}$$
$$\hline$$
$$size = 2048;\ level = 2$$

40

Z Formal Specification

Expressions and operators

- Expressions describe the values that variables might have.
- Expressions are formulas where names and literal values appear together with operators. Expressions are sometimes called terms.
  - Arithmetic expressions : 12 mid 5=2, a+b=12
  - Set expressions {1,2,3} ∪ {2,3,4} = {1,2,3,4}
- Expressions and types : Every expression has a type: the type of the value it denotes.
  - So every arithmetic expression has type Z, and every set expression has type $\mathbb{P}$ T, where T is the type of the set elements.

41

Z Formal Specification

Expressions and operators

- Erroneous expressions

DICE#	[SYNTAX ERROR! # is a prefix operator.]
12 div *red*	[TYPE ERROR! Second operand is not a number.]
{1, 2, 3} ∪ {*red*, *green*}	[TYPE ERROR! Operands are sets of different types.]

42