Z Specification:

Z specification is formal language used for modelling and describing system.It is based on mathematical notations, set theory, and lambda calculus

Formal specifications use mathematical notation to describe in a precise way the properties which an information system must have, without unduly constraining the way in which these properties are achieved

it describe what the system must do rather how?

This abstraction makes formal specifications useful in the process of developing a computer system. It is indepoendent of program code a formal specification of a system can be completed early in its development.The other main ingredient in Z is a way of decomposing a specification into small pieces called schemas.By splitting the specification into schemas, we can present it piece by piece. Each piece can be linked with a commentary which explains informally the significance of the formal mathematics. In Z, schemas are used to describe both static and dynamic aspects of a system. The static aspects include.

- state it can occupy
- invarient relationship that maintain system moves from state to state.

Dynamic aspect include.

- operations that are possible
- relationship between input and output
- the changes of state that happen

this model consist of description of state space and system operations.

System state space is a set of all states that the system could be in.

It describe the value of each variable.

We represent state space of a system as collection of functions, sets, relations, sequences, bags etc.

Together with collection of invarient properties on these objects.

It describe regularitiesbetween state changes.

Operations are generally defined in terms of pre and post conditions.

Operations define acceptable state transitions.

Z schemas.

It is a 2 dimentional graphical notation for describing.

1)state space and 2) operations.

Schema is either of the form

Schema name
Declararations
Perdicates

or

Schema name
Declarations

In the later case predicate part is assumed to be true
after this, schema name will be associated with schema which is content of box
declaration part of schema contain
list of variable declaration.
And reference to other schemas
variable declaration has usual form
x1,x2,x3,...,xn.T

predicate part of schema contaion list of predicates.separated by semicolens.

Use case formalization.
set of all unique identifiers throughout the specification is denoted by given set of classifier
there are several relationships between use cases. This include relationship, and generalization.
Include is a relationship between two use cases which is used to show behavior of the included use case is insserted into behavior of inluding use cases.

Exteend relationship is that specifies how and when behavior defined in usually operations extending use case can be inserted into behavior defined in the extended use case

The relationships include and extend are basically represented as a Z relation that relates two use cases. The relation include is used to record which use case is directly included by the original use case, whereas the relation extend is used to record which use case is directly extending the original use case.

The functions includedUsecases and extendingUsecases are respectively obtained with the transitive closure of include and extend. They are used to formally specify the function that returns the set of use cases that are directly or indirectly related with the relations include or extend.

The functions includedUsecases and extendingUsecases are respectively obtained with the transitive closure of include and extend. They are used to formally specify the function that returns the set of use cases that are directly or indirectly related with the relations include or extend.

Actor formalization.
An actor specifies a role played by a user or any other system that interacts with the subject
given set CLASSIFIER is used to define the set of all unique identifiers throughout the specification. The variable ACTOR is introduced as a classifier to specify the set of actors' identifiers.
The actor can be human user, some internal application or may be some external application. The actor type is introduced as an enumerated set representing two types of actors.
The relationship between actor and use case is basically represented as a Z relation that relates an actor to a use case. The relation association. AUis used to record which use case can be used by the actor, whereas the function relatedUsecases returns the set of use cases that are associated to the actor.
Actor generalization is used to factor out and reuse similarities between actors. This relationship shows that one actor inherits the role and properties of another actor. To represent generalization, we chose to add a state variable to declaration part of the schema Actor as shown below. This state variable is called parents and specified as a set of actors. The generalization relationship also implies that the descendant actor can use all the use cases that have been defined for its ancestor.

The specification of requirements is a key activity for achieving the goals of any software project and it has long been established and recognized by researchers and practitioners
The objectives of our work are providing concise and unambiguous specification of use case diagrams using Z notation and consistency checking by means of Z theorems

Formal specification of use case diagrams has been done in the past using different languages but we believe that the expressiveness and the mathematical basis of Z notation provide a richer framework for such formalism.fact that the use of formal specification can greatly enhance the quality of the produced software as well as to contribute significantly to cost savings in the software development process.
These are the connectives of propositional logic. they allow complex predicates to be built from simpler ones in such a way that the truth or falsity of the compound in some binding depends only on the truth or falsity of the parts in the same binding
For example, the predicate P1 ^ P2 is true in any binding if and only if both P1 and P2 are true in that binding.

Operations schemas

In specifying a system operation, we must consider
the objects that are accessed by the operation, and of these.
the objects that are known to remain unchanged by the operation (value parameters)
the objects that may be altered by the operation
the pre-conditions of the operation, i.e., the things that must be true for the operation to succeed.
the post-conditions the things that will be true after the operation, if the pre-condition was satisfied before the operation
Example telephone book.
Consider lookup operation
we put a name in, and get a phone number out
- this operation accesses the PhoneBook schema.
it does not change it.
- it takes a single ‘input’ a name for which we want to find a phone number.
it produces a single output a phone number
it has the pre-condition that the name is known to the database

Procedural Abstraction: Schemas only describe what is to be done; issues relating to “how” are ignored