

RE2: Assignment 4

Z explained

What is Z?

- Named after Zermelo-Fränkel set theory
- Formal specification language
- Used for describing and modelling computing systems.
- It is targeted at the clear specification of computer programs and the formulation of proofs about the intended program behavior.

Agenda

- Basics
 - Basic Types
 - Abbreviation Definitions
 - Axiom Descriptions
- Schemas
- Elements
- Example

Basics

- We can introduce a new **data type** by writing its name inside brackets:
 - [CHAR]

Basics

- We can introduce a new **data type** by writing its name inside brackets:
 - `[CHAR]`
- For defining **abbreviations**, we use the `==` operator:
 - `TEXT == seq CHAR`

Basics

- We can introduce a new **data type** by writing its name inside brackets:
 - [CHAR]
- For defining **abbreviations**, we use the == operator:
 - $TEXT == \text{seq CHAR}$
- **Axiom descriptions** consist of a *declaration* and a *predicate*
 - $$\frac{maxsize : N}{maxsize \leq 65535}$$

Basics

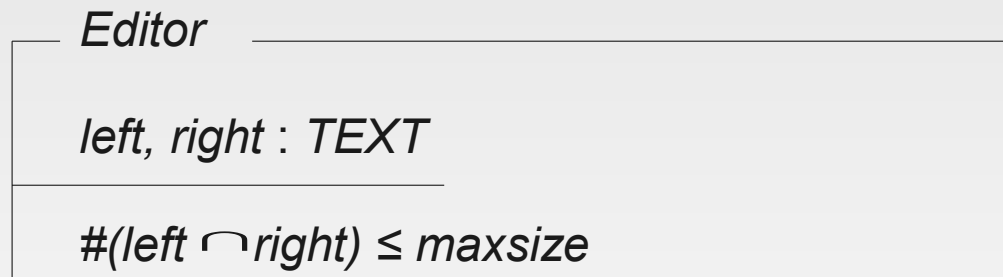
- Data types, abbreviations and axiom descriptions are **global**

Agenda

- Basics
- Schemas
 - State Schemas
 - Initialization Schemas
 - Operation Schemas
- Elements
- Example

State Schemas

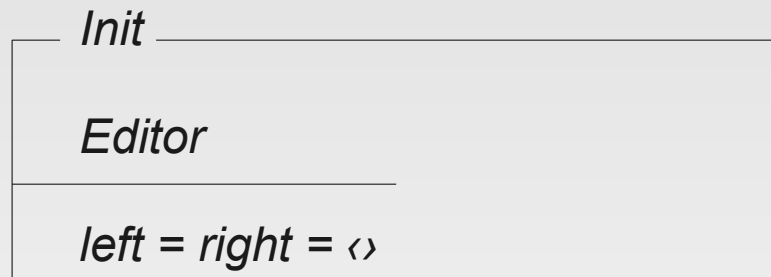
- Content of a systems memory
- No functionality



- Schema variables are local

Initialization Schemas

- *Init* is the start up schema



- By including the *Editor* schema in the declaration, we can access local state variables from *Editor*

Operation Schemas

- To model activity, Z provides the *operation schema*

printing : $\mathbb{P}$ CHAR

Insert

Δ *Editor*

ch? : CHAR

ch? $\in$ *printing*

left' = *left* $\frown$ *<ch?>*

right' = *right*

Agenda

- Basics
- Schemas
- Elements
 - Sets
 - Types
 - Variables
- Example

Sets

- One way to describe a set is to list its elements:
 - {red, yellow, green}
 - {1, 2, 3, 4, 5, 6}
- Element order is insignificant
- Some sets are built-in:
 - $\mathbb{Z}$ is the set of *integers*
 - $\mathbb{N}$ is the set of *natural numbers*
 - $\mathbb{N}_1$ is the set of *strictly positive integers*

Types

- All elements in a set must have the same *type*
- Every set has a *carrier set* that contains all of the objects of that type
- But not all sets are types
 - E.g. $\mathbb{N}$ is a set containing integers, but $\mathbb{Z}$ is the carrier set for integers
- Some set operations:
 - $A \cap B$ set intersection
 - $A \cup B$ set union
 - $A \setminus B$ set difference

Variables

- Variables can be declared using either a *type* or a *set*
 - $i : \mathbb{Z}$
 - $d_1, d_2 : 1 .. 6$
 - $signal : \{red, yellow, green\}$

Agenda

- Basics
- Schemas
- Elements
- Example
 - Document Control System

Document Control System

- We have a set of documents
- Users can check out a document in order to change it
- If a document is checked out, it is locked until it is checked in again (reading is still possible)
- Only users with appropriate permissions can check out a document

Document Control System

- Two basic sets:
 - [*PERSON*, *DOCUMENT*]
- Some people have permission to change particular documents:
 - | *permission* : *DOCUMENT* $\leftrightarrow$ *PERSON*
 - Relation as a set of pairs (*document*, *person*)

Document Control System

- Example data:

doug, aki, phil : PERSON
spec, design, code : DOCUMENT

permission = {(spec, doug), (design, doug), (design, aki), (code, aki), (code, phil)}

- System state

Documents

checked_out : DOCUMENT $\rightarrow$ PERSON

checked_out $\leq$ permission

Document Control System

- Operation to check out: *CheckOut*

CheckOut

$\Delta Documents$

$p? : PERSON$

$d? : DOCUMENT$

$d? \notin \text{dom } checked_out$

$(d?, p?) \in \text{permission}$

$checked_out' = checked_out \cup \{(d?, p?)\}$

Satisfying Preconditions

- The operation has two preconditions. We must account for cases where these are not satisfied:
 - already checked out:

$CheckedOut \triangleq [\exists Documents; d? : DOCUMENT \mid d? \in \text{dom } checked_out]$

- person does not have permission:

Unauthorized

$\exists Documents$

$p? : PERSON$

$d? : DOCUMENT$

$(d?, p?) \notin permission$

Total Operation

- The total operation $T_CheckOut$ covers all three possibilities
 - $T_CheckOut \triangleq CheckOut \vee CheckedOut \vee Unauthorized$