Module Interface Specification
MIS

SFWR ENG 2B03
2003
Robert L. Baber
MIS: Purpose

A Module Interface Specification (MIS)

- specifies the externally observable behaviour of a module’s access routines
- mathematically precisely
- input/output relationship
- domains, restrictions on use, “exceptions”
- application (“abstract”) state variables
- application, not implementation, oriented
MIS: Language

A Module Interface Specification is written in
- mathematical
- application language.

It is **not** written in the language of the implementation.
MIS: Viewpoint

A Module Interface Specification describes the intended behaviour of a module’s access routines from an
- external,
- outside viewpoint.

MIS: Viewpoint

A Module Interface Specification contains absolutely no information about the insides of the module or its access routines not implied by the specified external behaviour.

Internal aspects of the module’s implementation are secrets of the module.
MIS: Target audience

A Module Interface Specification is written for

- module designer and implementer
- inspectors, testers of the module
- designers and implementers of program segments using the module’s access routines

The last group above needs **only** the MIS.

MIS test: if not enough for them, then the MIS is not complete, not really an MIS
MIS: Content

- module name
- imported identifiers, e.g. constants, variables, data types, etc.
- exported identifiers, e.g. access routine names
- state variables
- state invariant
- assumptions
- access routine semantics
Access Routine Semantics: Content

- name
- input/output relation (e.g. precondition, postcondition)
- domain (e.g. restrictions, “exceptions”)
Access Routine Semantics: Content

- “input” consists of the values of the
  - formal parameters passed to the routine and
  - state variables before execution of the routine

- “output” consists of the values of the
  - formal parameters (and results, if any) passed by the routine back to the caller and
  - state variables after execution of the routine
The input/output relation

- restricts the pairs of “input” and “output” to reflect all requirements (the specification)
- must be complete, i.e. any routine satisfying the input/output relation is “correct” and acceptable
- refers to “output” and usually also “input”
Access Routine Semantics: Content

The domain (restrictions, “exceptions”)

- defines the set of “inputs” for which the routine must function
- refers only to the “input”, never to the “output”
- states exception(s) and condition(s) for triggering, when applicable
MIS Example: Stack Module

- name: Stack
- imported identifiers: A (data type, see below)
- exported access routines: init, push, pop, depth, full
- state variables: s, where s ∈ A* (s is a sequence of elements of A, A is any set)
- state invariant: |s| ≤ MaxDepth
- assumptions: init called before any other access routine
MIS Example: Stack Module

where MaxDepth

- a positive integer
- an internal implementation parameter
- value is a design secret (secret at design time)
MIS Example: Access Routine Semantics

- **name**: init
- **input/output relation**: $s' = [ ]$
- **restrictions on use**: none
MIS Example: Access Routine Semantics

- **name:** push(x)
- **input/output relation:** $s' = s \land [x]$
- **domain:** $(|s| < \text{MaxDepth}) \land (x \in A)$
- **alternative:**
  
  $(|s| < \text{MaxDepth}) \land (s' = s \land [x])$
  
  $\lor (|s| = \text{MaxDepth}) \land (s' = s)$
MIS Example: Access Routine Semantics

- name: pop
- input/output relation:
  
  \[ ('s = (s' \& \ [result])) \land (result \in A) \]
- restrictions on use: \( 0 < |s| \)
MIS Example: Access Routine Semantics

- name: depth
- input/output relation: \( \text{result} = |'s'| \land (s' = 's) \)
- restrictions on use: none
MIS Example: Access Routine Semantics

- name: full
- input/output relation:
  \[ result = (|s| = \text{MaxDepth}) \land (s' = s) \]
- restrictions on use: none
MIS Example: Access Routine Semantics

Usual convention:

- If any state variable is not explicitly mentioned in the input/output relation, its value is not changed by the access routine in question.
MIS: Summary

Module Interface Specification

- external view
- mathematically precise
- application ("abstract") state variables
- application, not implementation, oriented
- not in the language of the implementation, but
- in mathematical and application language