Module Interface Specification

Michael Balas

January 4, 2019

Overview of MIS

The Module Interface Specifications (MIS) precisely specifies the module's observable behaviour (i.e. *what it does*), though it does not specify internal design. This idea is inspired from the principles of software engineering.

MIS Template

Uses

- Specifies imported constants, data types and access programs. The specification of one module will often depend on using the interface specified by another module. When there are many modules the *Uses* information is very useful for navigation of the documentation.

Syntax

- Specifies exported constants and types, as well as access routine names with input and output parameter types and exceptions. Access routines are shown in a tabular format.

Semantics

- Specifies state variables (which define the state space) and state invariants (predicates on the state space that restrict the *legal* states of the module). After every access routine call, the state should satisfy the invariant. Local functions, types and constants are also declared for specification purposes only, as well as any relevant considerations (for information that does not fit elsewhere).

Point ADT Module

Template Module

pointADT

Uses

N/A

Syntax

Exported Types

PointT = ?

Exported Access Programs

Routine name	In	Out	Exceptions
new PointT	real, real	PointT	
xcrd		real	
ycrd		real	
dist	PointT	real	
rot	real		

Semantics

State Variables

xc: real
yc: real

State Invariant

None

Assumptions

None

Access Routine Semantics

new PointT (x, y):

- transition: $xc, yc := x, y$
- output: $out := self$
- exception: none

xcrd:

- output: $out := xc$
- exception: none

ycrd:

- output: $out := yc$
- exception: none

dist(p):

- output: $out := \sqrt{(xc - p.\text{xcrd}())^2 + (yc - p.\text{ycrd}())^2}$
- exception: none

rot(ϕ):

- ϕ is in radians
- transition:

$$\begin{bmatrix} xc \\ yc \end{bmatrix} := \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} xc \\ yc \end{bmatrix}$$

- exception: none

Line Module

Template Module

lineADT

Uses

pointADT

Syntax

Exported Types

LineT = ?

Exported Access Programs

Routine name	In	Out	Exceptions
new LineT	PointT, PointT	LineT	
beg		PointT	
end		PointT	
len		real	
mdpt		PointT	
rot	real		

Semantics

State Variables

b: PointT
e: PointT

State Invariant

None

Assumptions

None

Access Routine Semantics

new LineT (p_1, p_2):

- transition: $b, e := p_1, p_2$
- output: $out := self$
- exception: none

beg:

- output: $out := b$
- exception: none

end:

- output: $out := e$
- exception: none

len:

- output: $out := b.\text{dist}(e)$
- exception: none

mdpt:

- output:

$$out := \text{new PointT}(\text{avg}(b.\text{xcrd}(), e.\text{xcrd}()), \text{avg}(b.\text{ycrd}(), e.\text{ycrd}()))$$

- exception: none

rot (ϕ):

- ϕ is in radians
- transition: $b.\text{rot}(\phi), e.\text{rot}(\phi)$
- exception: none

Local Functions

avg: real $\times$ real $\rightarrow$ real
$\text{avg}(x_1, x_2) \equiv \frac{x_1+x_2}{2}$

Circle Module

Template Module

circleADT

Uses

pointADT, lineADT

Syntax

Exported Types

CircleT = ?

Exported Access Programs

Routine name	In	Out	Exceptions
new CircleT	PointT, real	CircleT	
cen		PointT	
rad		real	
area		real	
intersect	CircleT	boolean	
connection	CircleT	LineT	
force	real $\rightarrow$ real	CircleT $\rightarrow$ real	

Semantics

State Variables

c: PointT
r: real

State Invariant

None

Assumptions

None

Access Routine Semantics

new CircleT (cin, rin):

- transition: $c, r := cin, rin$
- output: $out := self$
- exception: none

cen:

- output: $out := c$
- exception: none

rad:

- output: $out := r$
- exception: none

area:

- output: $out := \pi r^2$
- exception: none

intersect(ci):

- output: $\exists(p : \text{PointT} | \text{insideCircle}(p, ci) : \text{insideCircle}(p, self))$
- exception: none

connection(ci):

- output: $out :=$ new LineT$(c, ci.cen())$
- exception: none

force(f):

- output: $out := \lambda x \rightarrow self.\text{area}() \cdot x.\text{area}() \cdot f(self.\text{connection}(x).\text{len}())$
- exception: none

Local Functions

insideCircle: PointT $\times$ CircleT $\rightarrow$ boolean
insideCircle$(p, c) \equiv p.\text{dist}(c.\text{cen}()) \leq c.\text{rad}()$

Deque Of Circles Module

Module

DequeCircleModule

Uses

circleADT

Syntax

Exported Constants

MAX_SIZE = 20

Exported Access Programs

Routine name	In	Out	Exceptions
Deq_init			
Deq_pushBack	CircleT		FULL
Deq_pushFront	CircleT		FULL
Deq_popBack			EMPTY
Deq_popFront			EMPTY
Deq_back		CircleT	EMPTY
Deq_front		CircleT	EMPTY
Deq_size		integer	
Deq_disjoint		boolean	EMPTY
Deq_sumFx	real $\rightarrow$ real	real	EMPTY
Deq_totalArea		real	EMPTY
Deq_averageRadius		real	EMPTY

Semantics

State Variables

s: sequence of CircleT

State Invariant

$|s| \leq$ MAX_SIZE

Assumptions

Deq_init() is called before any other access program.

Access Routine Semantics

Deq_init():

- transition: $s :=<>$
- exception: none

Deq_pushBack(c):

- transition: $s := s|| < c >$
- exception: $exc := (|s| = \text{MAX\_SIZE} \Rightarrow \text{FULL})$

Deq_pushFront(c):

- transition: $s :=< c > ||s$
- exception: $exc := (|s| = \text{MAX\_SIZE} \Rightarrow \text{FULL})$

Deq_popBack():

- transition: $s := s[0..|s| - 2]$
- exception: $exc := (|s| = 0 \Rightarrow \text{EMPTY})$

Deq_popFront():

- transition: $s := s[1..|s| - 1]$
- exception: $exc := (|s| = 0 \Rightarrow \text{EMPTY})$

Deq_back():

- output: $out := s[|s| - 1]$
- exception: $exc := (|s| = 0 \Rightarrow \text{EMPTY})$

Deq_front():

- output: $out := s[0]$
- exception: $exc := (|s| = 0 \Rightarrow \text{EMPTY})$

Deq_size():

- output: $out := |s|$
- exception: none

Deq_disjoint():

- output

$$out := \forall(i, j : \mathbb{N} | i \in [0..|s| - 1] \wedge j \in [0..|s| - 1] \wedge i \neq j : \neg s[i].\text{intersect}(s[j]))$$

- exception: $exc := (|s| = 0 \Rightarrow \text{EMPTY})$

Deq_sumFx(f):

- output

$$out := +(i : \mathbb{N} | i \in ([1..|s| - 1]) : \text{Fx}(f, s[i], s[0]))$$

- exception: $exc := (|s| = 0 \Rightarrow \text{EMPTY})$

Deq_totalArea():

- output

$$out := +(i : \mathbb{N} | i \in ([0..|s| - 1]) : s[i].area())$$

- exception: $exc := (|s| = 0 \Rightarrow \text{EMPTY})$

Deq_averageRadius():

- output

$$out := +(i : \mathbb{N} | i \in ([0..|s| - 1]) : \frac{s[i].rad()}{|s|})$$

- exception: $exc := (|s| = 0 \Rightarrow \text{EMPTY})$

Local Functions

Fx: (real $\rightarrow$ real) $\times$ CircleT $\times$ CircleT $\rightarrow$ real
$\text{Fx}(f, ci, cj) \equiv \text{xcomp}(ci.\text{force}(f)(cj), ci, cj)$

xcomp: real $\times$ CircleT $\times$ CircleT $\rightarrow$ real

$$\text{xcomp}(F, ci, cj) \equiv F\left[\frac{ci.\text{cen}().\text{xcrd}() - cj.\text{cen}().\text{xcrd}()}{ci.\text{connection}(cj).\text{len}()}\right]$$