# Software Spec – IRadialMesh Interface

Here I want to use the strategy that RadialMesh contains a mesh, but an IEnumerator is used to walk it. This allows multiple asynchronous routines to share a common (read-only) mesh construct. This Enumerator is special in that it needs methods to access points on either side of the currently selected mesh point.

Public Interface IRadialMesh

Inherits IEnumerable

Function X(index as Integer) as Double

* Should be sorted such that X(i+1) is always greater than X(i)
* X(i+1) = X(i) is not allowed. Every point should be greater than the last
* Attempt to access X(index < 0) or X(index >= count) should throw an IndexOutOfRangeException

Function GetEnumerator() as IEnumerator Implements IEnumerable.GetEnumerator

* Returns an instance of RadialMeshEnumerator

Readonly Property Type() as MeshType

* Returns a value of the MeshType enumeration indicating the specific type of mesh instance.

Readonly Property Count() as Integer

* Returns the number of points in the mesh.

Function DX(startIndex as Integer, endIndex as Integer) as Double

* Returns X(endIndex) – X(startIndex)
* If endIndex < startIndex, returns a negative value.
* If startIndex = endIndex, returns 0.0
* If startIndex is < 0 or endIndex > Count-1, throws an IndexOutOfRangeException

Function Range() as Double

* Returns X(Count-1) – X(0), the total range of the mesh.

Function Max() as Double

* Returns X(Count-1)

Function Min() as Double

* Returns X(0)

Function BlockCount() as Integer

* Returns the number of blocks in a HermanSkillman mesh. For other (non-block based) meshes, returns 1.

MeshType Enumeration

HermanSkillman = 0

Cubic = 2

Spec for RadialMeshEnumerator Class

This class walks an IRadialMesh

Implements IEnumerator<Of Double>

Readonly Property Current() as Double Implements IEnumerator.Current

* Return the value of the current point X(currentIndex)

Function MoveNext() as Boolean Implements IEnumerator.MoveNext

* Increments the CurrentIndex pointer.
* When called at end of list, throws an InvalidOperationException.
* Returns FALSE if pointer is at the end of the list, TRUE otherwise.

Function MovePrevious() as Boolean

* Decrements the CurrentIndex pointer.
* When called at the beginning of the list, throws an InvalidOperationException.
* Returns FALSE if pointer is at the beginning of the list, TRUE otherwise.

Sub SetToBeginning() Implements IEnumerator.Reset

* Sets the CurrentIndex pointer at -1 (enumerators are positioned before the first element until MoveNext is called)

Sub SetToEnd()

* Sets the CurrentIndex pointer at Count (positioned after the last element until MovePrevious is called)

Function DXPlus() as Double

* Returns X(currentIndex + 1) – X(currentIndex)
* If currentIndex > Count-1, then throw an InvalidOperationException.
* If called when currentIndex = -1, then throw an InvalidOperationException.

Function DXMinus() as Double

* Returns X(currentIndex) – X(currentIndex – 1)
* If currentIndex < 1, then throw an InvalidOperationException.
* If called when currentIndex = Count, then throw an InvalidOperationException.

Function DXPlus2() as Double

* Returns X(currentIndex + 2) – X(currentIndex + 1)
* If currentIndex > Count-2, then throw an InvalidOperationException.
* If called when currentIndex = -1, then throw an InvalidOperationException.

Function DXMinus2() as Double

* Returns X(currentIndex – 1) – X(currentIndex – 2)
* If currentIndex < 2, then throw an InvalidOperationException.
* If called when currentIndex = Count, then throw an InvalidOperationException.

Function CurrentIndex() as Integer

Implementation for the Herman-Skillman mesh

Public Class HermanSkillmanMesh

Implements IRadialMesh

Sub New(Z as Element, deltaX as Double, size as HermanSkillmanMeshSize)

* deltaX must be greater than zero.

Sub New(Z as Element, size as HermanSkillmanMeshSize)

* Assumes default deltaX.

Sub New(Z as Element)

* Assumes a small (441 point) mesh and deltaX = 0.0025 (bohrs)

Readonly Property CMU() as Double

* Returns CMU = 0.5 \* Pow(3 \* Pi / 4, 2/3) \* Pow(Z, -1/3)

Readonly Property Z() as Element

* Returns the atomic number (Element enumeration)

Readonly Property DeltaX() as Double

* Returns the (initial) deltaX parameter

Readonly Property Size() as HermanSkillmanMeshSize

* Returns the size of the mesh (441, 481, or 521 points)

Public Enum HermanSkillmanMeshSize

Small = 441

Medium = 481

Large = 521