LexTech

AFGROW .NET

AFGROW Workshop 2023

AFGROW COM Interface New and Upcoming Features

James Lambert, LexTech, Inc.

LexTech

Overview

AFGROW .NET

- API 579 solutions through the AFGROW COM Interface
- Plugin models through AFGROW COM Interface
- Multi Site Damage through the AFGROW COM Interface
- *Upcoming* Extended Event Reporting
- List of New and Upcoming COM features
- How To: Release AFGROW COM Objects in C# and VB (.NET)
- Other General Changes and Bug Fixes

API 579 Models and Special Cases

- With AFGROW 5.4 came the addition of 22 new pipeline models that are documented in “Fitness-For-Service, API 579-1/ASME FFS-1”
- These include pressure, weight function, and polynomial based solutions for several geometries
- Each of these new models may be run with AFGROW’s automation capabilities.

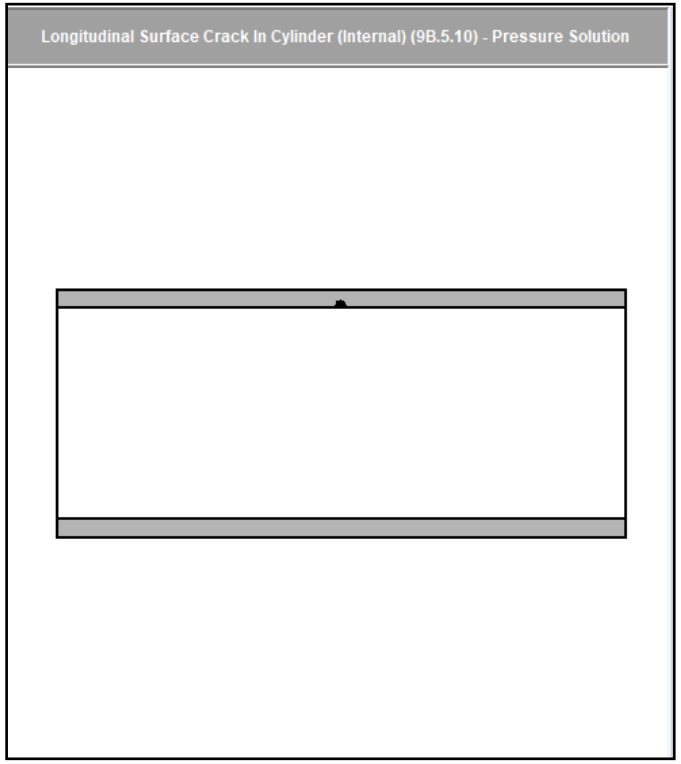
API 579 Pressure Models

- Pressure based models use an internal pressure reference rather than axial stress
- This internal pressure is modified using the input spectrum and the SMF
- SetCylinderProp() is used for models with a width value
- SetPipeProp() is used to set models without a width value

	Description of Configuration	Beta Solution	Model Code	API 579 Code
	Longitudinal Through Crack in Cylinder	Pressure Load (API 579)	5010	9B.5.1
	Circumferential Through Crack in Cylinder	Pressure Load (API 579)	5030	9B.5.3
	Infinite Longitudinal Surface Crack in Cylinder (Internal)	Pressure Load (API 579)	5040	9B.5.4
	Infinite Longitudinal Surface Crack in Cylinder (External)	Pressure Load (API 579)	5041	9B.5.4
	Circumferential Crack in Cylinder (Internal)	Pressure Load (API 579)	5070	9B.5.7
	Circumferential Crack in Cylinder (External)	Pressure Load (API 579)	5071	9B.5.7
	Longitudinal Surface Crack in Cylinder (Internal)	Pressure Load (API 579)	5100	9B.5.10
	Longitudinal Surface Crack in Cylinder (External)	Pressure Load (API 579)	5101	9B.5.10
	Circumferential Surface Crack in Cylinder (Internal)	Pressure Load (API 579)	5130	9B.5.13
	Circumferential Surface Crack in Cylinder (External)	Pressure Load (API 579)	5131	9B.5.13

API 579 Pressure Models Code Sample

```
Afgrow.Application af = new Afgrow.Application();

af.Model = Afgrow.AfgrowModels.aLongInternalSurfaceInCyl;

af.SetCylinderProp(8.0, 3.0, 3.5);

af.ConstAmplitudeSpectrum(0.8);
af.SMF = 1.5; //Represents max pressure in ksi for constant amplitude spectrum

af.PredictInfoExt += Af_PredictInfoExt;

af.RunPredict();

Console.ReadLine();
```



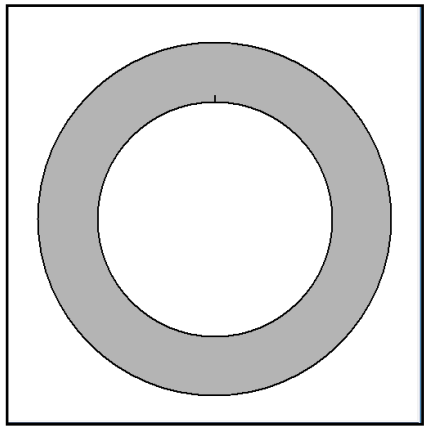
API 579 Weight Function Models

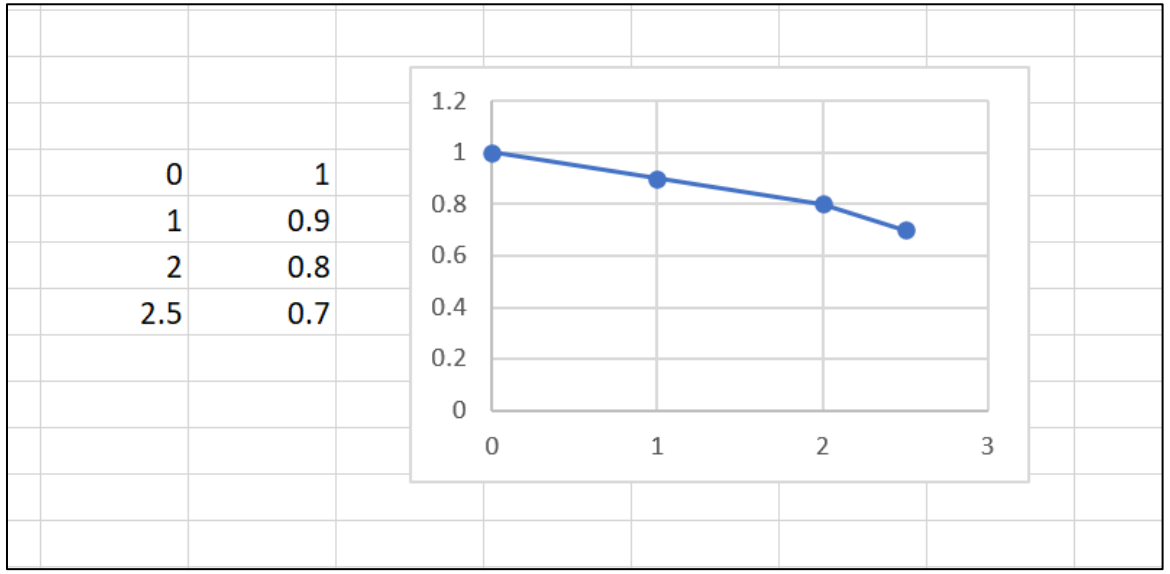
- API 579 Weight Function models use a normal stress reference at the crack origin
- When setting stress distributions with the function SetWFStressDistr(), the parameter CXdirection should be used for through cracks, and the parameter AYdirection should be used for part-through cracks
- For these models, the stress distribution is always in the thickness (y) direction

	Description of Configuration	Beta Solution	Model Code	API 579 Code
	Longitudinal Surface Crack in Cylinder (Internal)	User Defined	5060	9B.5.6
	Longitudinal Surface Crack in Cylinder (External)	Application Defined	5061	9B.5.6
	Circumferential Crack in Cylinder (Internal)	Application Defined	5090	9B.5.9
	Circumferential Crack in Cylinder (External)	Application Defined	5091	9B.5.9
	Longitudinal Surface Crack in Cylinder (Internal)	Application Defined	5120	9B.5.12
	Longitudinal Surface Crack in Cylinder (External)	Application Defined	5121	9B.5.12

API 579 Weight Function Models Code Sample

```
Afgrow.Application? af = new Afgrow.Application();
AdvancedModelInt? advModel = null;
advModel = af.AdvancedModel;

af.Model = Afgrow.AfgrowModels.wInfLongInternalSurfaceInCyl;

af.SetPipeProp(10.0, 15.0);
af.CrackLengthC = 0.3;

double[,] sd = new double[4, 2];
sd[0, 0] = 0.0;
sd[0, 1] = 1.0;
sd[1, 0] = 1.0;
sd[1, 1] = 0.9;
sd[2, 0] = 2.0;
sd[2, 1] = 0.8;
sd[3, 0] = 2.5;
sd[3, 1] = 0.7;


af.SetWFStressDistr(Afgrow.CrackDirection.CXdirection, sd);

af.ConstAmplitudeSpectrum(0.0);
af.SMF = 5;

double cycles;
af.RunFrozPredict(out cycles, out _, out _, out _, out _, out _, out _);
```





Plot of sample stress distribution

API 579 Polynomial Models

- Polynomial based models use a 4th order polynomial to define a stress distribution starting from the crack origin
- The polynomial coefficients can be set using SetPolynomialStresses()

boolean SetPolynomialStresses(double s0, double s1, double s2, double s3, double s4)

Description

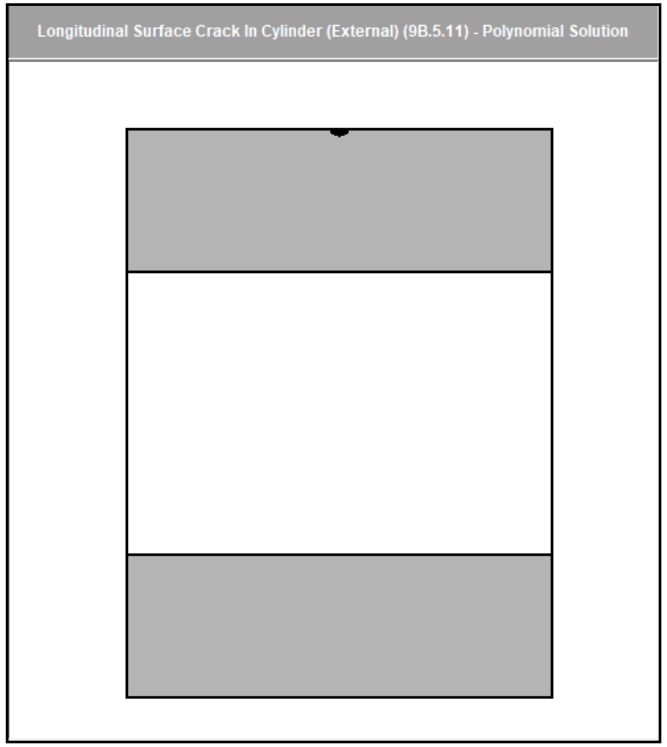
Sets the polynomial coefficients for the polynomial API 579 Models. The polynomial equation takes this form:

$$s_0 + s_1 * x + s_2 * x^2 + s_3 * x^3 + s_4 * x^4$$

	Description of Configuration	Beta Solution	Model Code	API 579 Code
	Longitudinal Surface Crack in Cylinder (Internal)	User Defined	5110	9B.5.11
	Longitudinal Surface Crack in Cylinder (External)	Application Defined	5111	9B.5.11
	Circumferential Surface Crack in Cylinder (Internal)	Application Defined	5140	9B.5.14
	Circumferential Surface Crack in Cylinder (External)	Application Defined	5141	9B.5.14
	Longitudinal Full-Elliptic Embedded Crack in Cylinder	Application Defined	5180	9B.5.18
	Full-Elliptic Embedded Crack in Cylinder	Application Defined	5190	9B.5.18

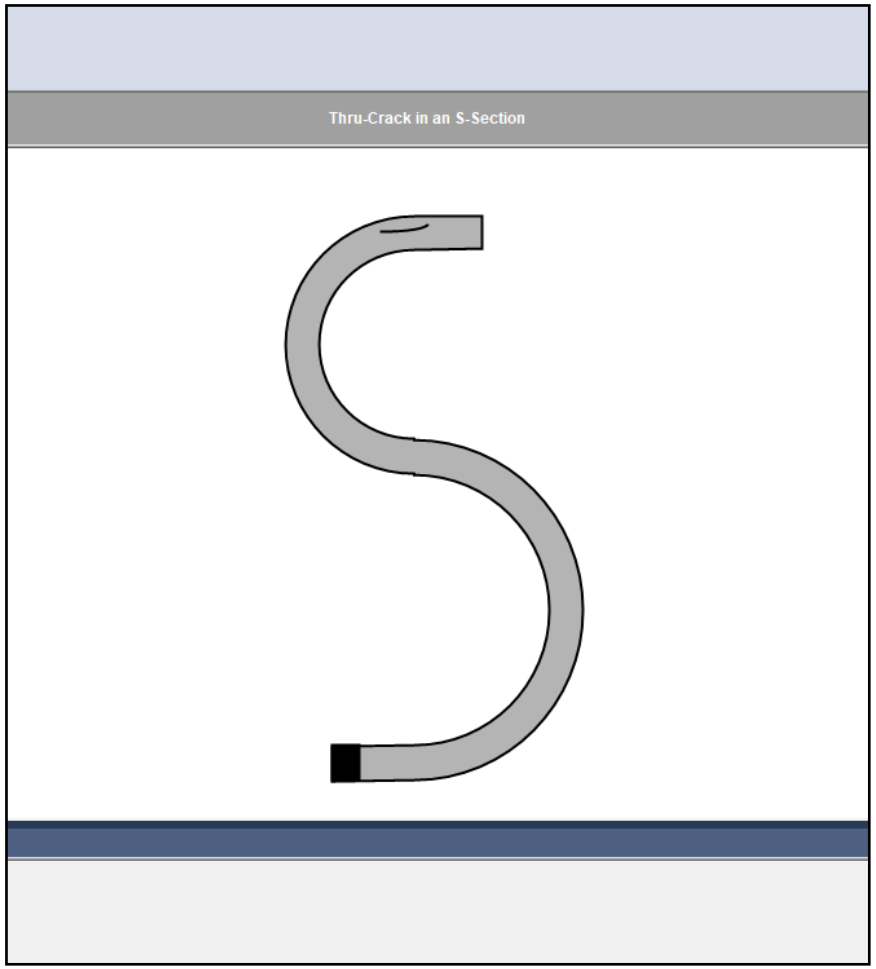
API 579 Polynomial Code Sample

```
Dim af = New Afgrow.Application()

af.Model = Afgrow.AfgrowModels.pLongExternalSurfaceInCyl

af.SetCylinderProp(15.0, 10.0, 20.0)
af.CrackLengthC = 0.2
af.CrackLengthA = 0.15

af.SetPolynomialStresses(1.050505051, -1.086367677, 14.6370101, -121.8371717, 237.8347475)

af.ConstAmplitudeSpectrum(0.0)
af.SMF = 14

Dim cycles As Double
Dim d As Double

'af.Visible = True
af.RunFrozPredict(cycles, d, d, d, d, d, d)
Console.WriteLine("Cycles: {0}", cycles)
```



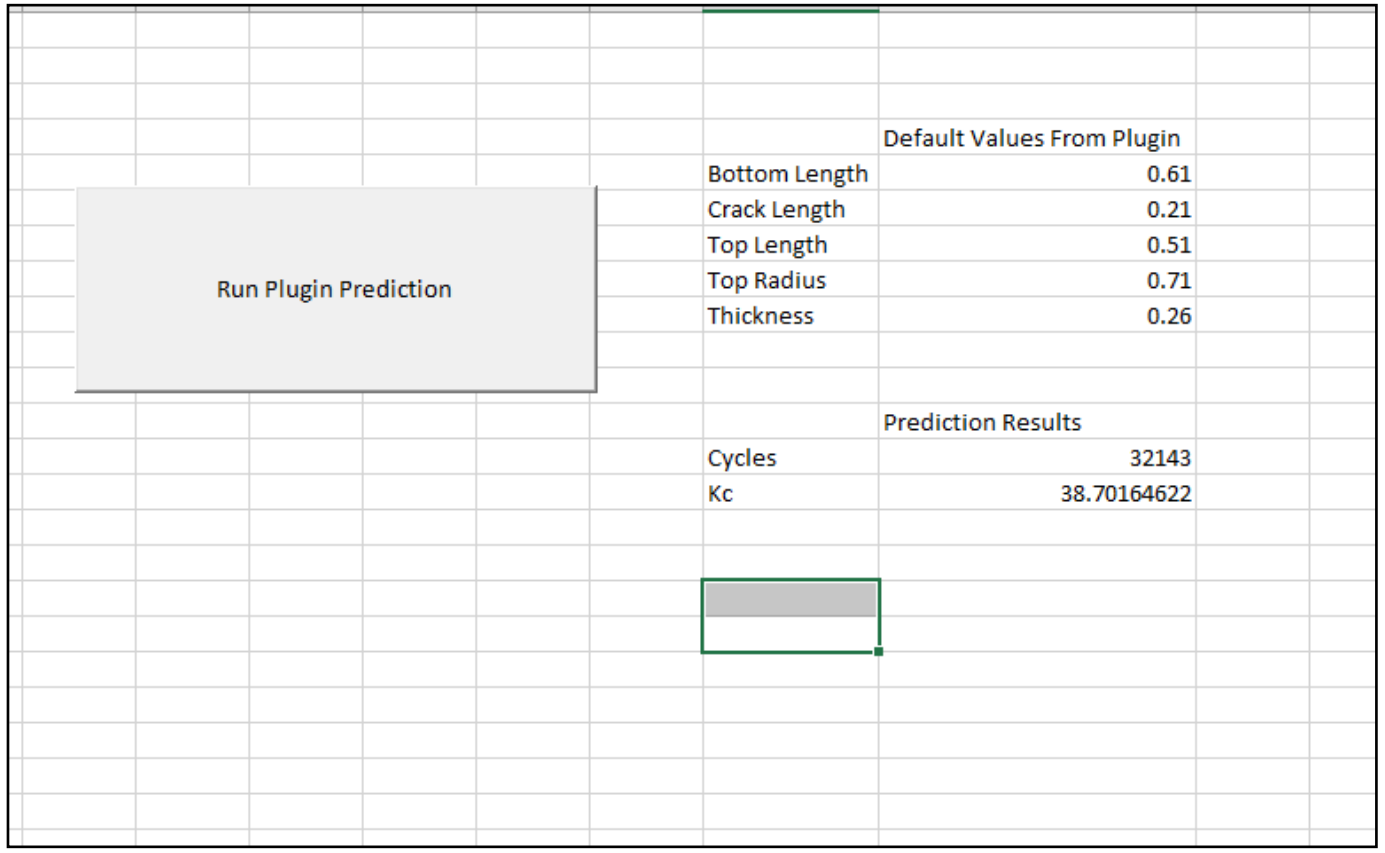
Plugin Models

- Plugin Models can now be run through AFGROW's automation capability
- Plugins can be loaded through their class name listed in AFGROW's configuration file
- Plugin properties can be set using the corresponding property label



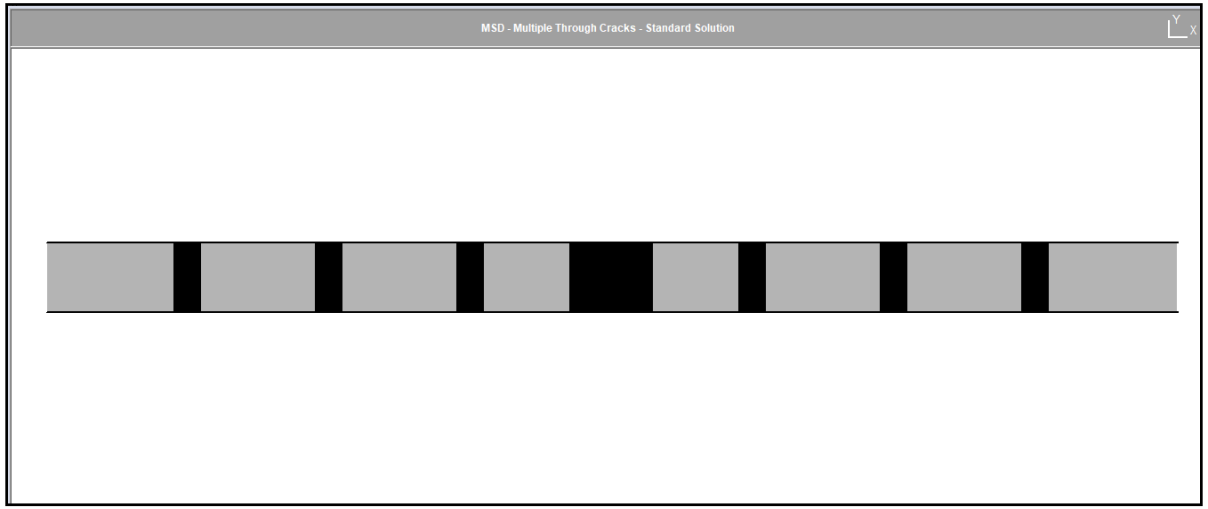
Plugin Model Code Sample

```
Dim af As Afgrow.Application
Set af = CreateObject("Afgrow.Application")

af.Visible = True

Dim pl As Afgrow.PluginModelInt
Set pl = af.PluginModel

'same label as located in the afgrow.exe.config file
pl.LoadPlugin "VZLUPlugin.VZLUPluginClass"

pl.SetProperty "Bottom Length", Cells(5, 10).Value
pl.SetProperty "Crack Length", Cells(6, 10).Value
pl.SetProperty "Top Length", Cells(7, 10).Value
pl.SetProperty "Top Radius", Cells(8, 10).Value
pl.SetProperty "Thickness", Cells(9, 10).Value

af.ConstAmplitudeSpectrum 0#
af.SMF = 14


Dim cycles As Double, kc As Double, d As Double
af.RunFrozPredict cycles, d, kc, d, d, d, d

Cells(13, 10) = cycles
Cells(14, 10) = kc

af.Quit
```



Multi Site Damage Models

- The new Multi Site Damage solutions are available through the COM Interface
- New functions have been implemented for setting model parameters
- Results for MSD predictions may be accessed through AFGROW's new Extended Reporting Events



LexTech

Multi Site Damage Models

AFGROW.NET

```
Dim af As Afgrow.Application
Set af = CreateObject("Afgrow.Application")

af.Visible = True

af.Model = aMultipleThroughCracksWithHoles
af.SetMSDPropWithHole 0.04, 0.03, 1.2, 5, 0.65

af.ConstAmplitudeSpectrum 0#
af.SMF = 14

Dim d As Double
Dim cycles As Double

af.RunFrozPredict cycles, d, d, d, d, d, d
```

Extended Reporting Events

- With AFGROW 5.5, there will be new reporting events to completely enumerate every crack front for every model
- Previously, for events like “PredictFinished” and “PredictInfo” there were 4 interfaces to access crack front information
- With the new events, information for every crack front will be available through a COM SafeArray, which can be accessed through COM capable languages.

```
[id(6)] void PredictFinishedExt(short result, double dblCycles, VARIANT* iOutputArray);
[id(7)] void PredictInfoExt(AfgrowModels model, double dStress, double dRStress, double dCycles, long dPass, VARIANT* iOutputArray);
[id(8)] void TransitionInfoExt(long nType, long nReason, double dStress,  double dRStress, double dCycles, long dPass, VARIANT* iOutputArray);
[id(9)] void FractureInfoExt(long nReason, double dStress, double dRStress, double dCycles, long dPass, VARIANT* iOutputArray);
```

LexTech

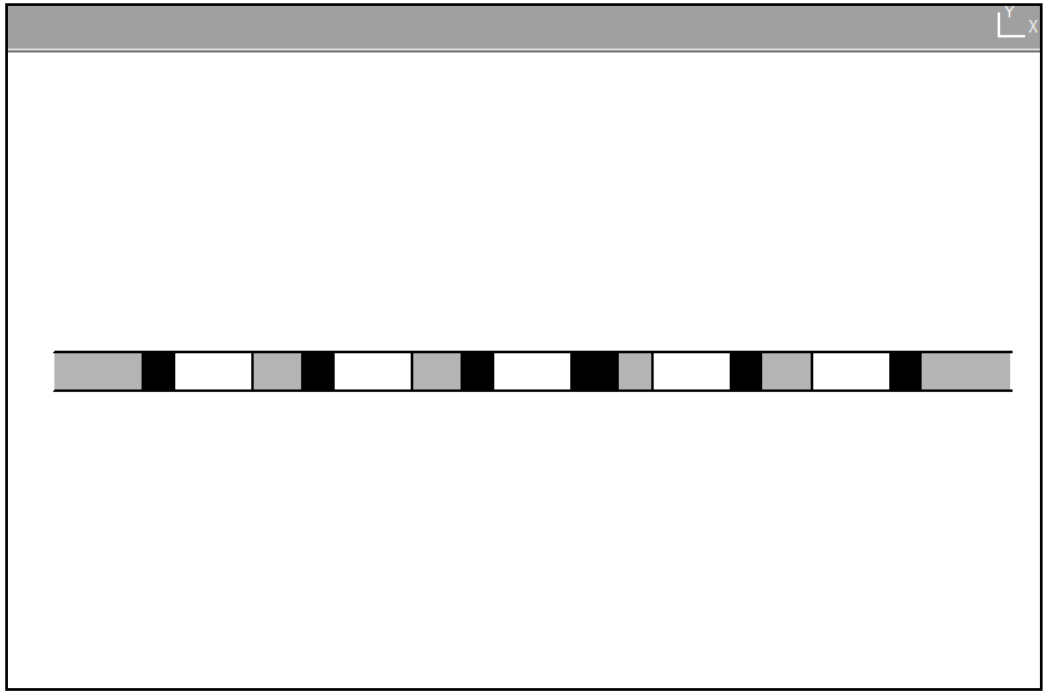
Extended Reporting Code Sample

AFGROW.NET

```
1 reference | Jimmy, 54 days ago | 1 author, 2 changes
private static void Af_PredictInfoExt(AfgrowModels Model, double dStres
{
    object[]? arr = iOutputArray as object[];

    for (int i = 0; i < arr.Length; i++)
    {
        IOutputInfoInt outputInfo = (IOutputInfoInt)arr[i];
        Console.WriteLine("Predict Info Extended ({0}, {1}): {2}, {3},
    }
}
```

Extended Reporting Results

```
Predict Info Extended (cycles, Key): Crack-Length, Crack-ID, Hole-ID, Direction-ID
Predict Info Extended (0, CS1): 0.2, 0, 0, 0
Predict Info Extended (0, CS2): 0.2, 1, 1, 0
Predict Info Extended (0, C11): 0.2, 2, 2, 0
Predict Info Extended (0, C12): 0.3, 2, 2, 1
Predict Info Extended (0, CS4): 0.2, 3, 3, 0
Predict Info Extended (0, CS5): 0.2, 4, 4, 0

Predict Finished Extended (Key): Result, Cycles, Crack-Length, Crack-ID, Hole-ID, Direction-ID
Predict Finished Extended (CS1): 1, 0, 0.2, 0, 0, 0
Predict Finished Extended (CS2): 1, 0, 0.2, 1, 1, 0
Predict Finished Extended (C11): 1, 0, 0.2, 2, 2, 0
Predict Finished Extended (C12): 1, 0, 0.3, 2, 2, 1
Predict Finished Extended (CS4): 1, 0, 0.2, 3, 3, 0
Predict Finished Extended (CS5): 1, 0, 0.2, 4, 4, 0
```

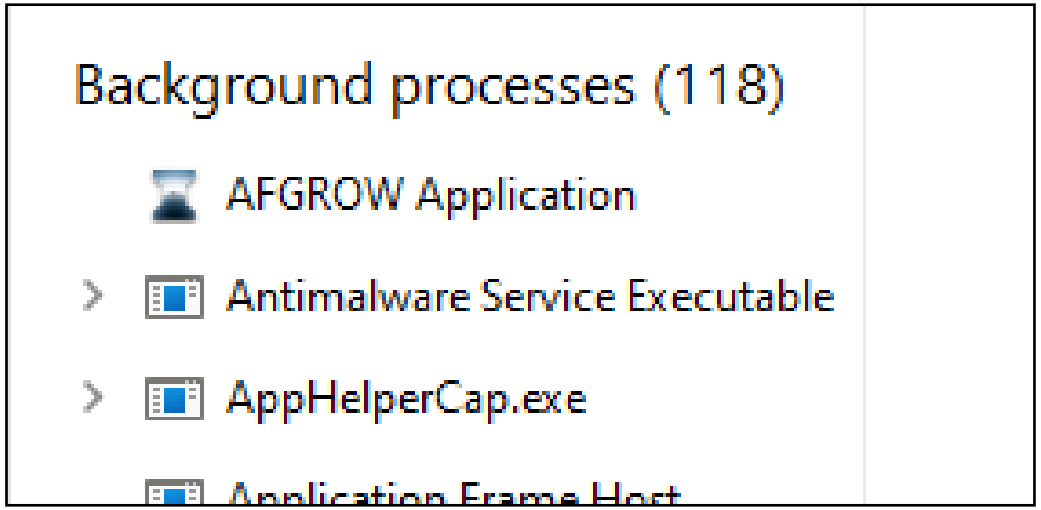


LexTech

Hanging or Crashed AFGROW in Memory

AFGROW .NET

- When a COM Application calls AFGROW to make a prediction, AFGROW has the potential to hang in memory.
- This may be seen in the “Background Processes” tab of Task Manager
- This may cause issues when running predictions



Potential Cause and Solutions for Crashed or Hanging AFGROW Processes

- Not releasing or improperly releasing the AFGROW COM Objects can cause AFGROW to hang after being ran through COM
- In C# and VB (.NET) AFGROW objects should be released manually, and the Garbage collector should be called
- When these steps are taken, AFGROW can close correctly and avoid inconsistent behaviors or crashes

LexTech

Closing AFGROW in .NET Code Sample

AFGROW .NET

```
af.Quit();

if (advModel != null) Marshal.ReleaseComObject(advModel);
if (af != null) Marshal.ReleaseComObject(af);

af = null;

GC.Collect();
GC.WaitForPendingFinalizers();
GC.Collect();
GC.WaitForPendingFinalizers();
```

LexTech

New Functions, Events, and Properties

AFGROW.NET

```
//General Properties
double CrackFacePressure;
IDispatch* PluginModel;
boolean WFUseXDistribution;
double CrackLengthCSecondary;
double MSDDistanceBetweenObjects;
int MSDNumberOfObjects;
//General Function
boolean SetPolynomialStresses(double s0, double s1, double s2, double s3, double s4);
boolean GetPolynomialStresses([out] double* s0, [out] double* s1, [out] double* s2, [out] double* s3, [out] double* s4);
boolean SetMSDProp(double CPrimary, double CSecondary, double distanceBetweenCracks, int numberOfCracks);
boolean SetMSDPropWithHole(double CPrimary, double CSecondary, double distanceBetweenHoles, int numberOfHoles, double holeDiamater);
//Predict Property Properties
boolean bSynchronizeOutput;
boolean bUseDefaultBendingCorrectionFactor;
double dBendingCorrectionFactor;
boolean bUseHarterFWCorrection;
//Extended Result Events
void PredictFinishedExt(short result, double dblCycles, VARIANT* iOutputArray);
void PredictInfoExt(AfgrowModels model, double dStress, double dRStress, double dCycles, long dPass, VARIANT* iOutputArray);
void TransitionInfoExt(long nType, long nReason, double dStress,  double dRStress, double dCycles, long dPass, VARIANT* iOutputArray);
void FractureInfoExt(long nReason, double dStress, double dRStress, double dCycles, long dPass, VARIANT* iOutputArray);Z
```

LexTech

New Models

AFGROW.NET

```
wSingleThroughAtHole            = 4032
wDoubleThroughAtHole            = 4034
aLongThroughInCyl               = 5010
aCircThroughInCyl               = 5030
aInfLongInternalSurfaceInCyl    = 5040
aInfLongExternalSurfaceInCyl    = 5041
wInfLongInternalSurfaceInCyl    = 5060
wInfLongExternalSurfaceInCyl    = 5061
aInternalCircInCyl              = 5070
aExternalCircInCyl              = 5071
wInternalCircInCyl              = 5090
wExternalCircInCyl              = 5091
aLongInternalSurfaceInCyl       = 5100
aLongExternalSurfaceInCyl       = 5101
pLongInternalSurfaceInCyl       = 5110
pLongExternalSurfaceInCyl       = 5111
wLongInternalSurfaceInCyl       = 5120
wLongExternalSurfaceInCyl       = 5121
aCircInternalSurfaceInCyl       = 5130
aCircExternalSurfaceInCyl       = 5131
pCircInternalSurfaceInCyl       = 5140
pCircExternalSurfaceInCyl       = 5141
pLongEllipticEmbeddedInCyl      = 5180
pCircEllipticEmbeddedInCyl      = 5190
aMultipleThroughCracks          = 9010
aMultipleThroughCracksWithHoles = 9020
```

LexTech

Questions?

AFGROW.NET