Deere Unit of Measure system

- XML file contains definition of units, the scale, offset, name, label and localization
- The unit system is grouped by UnitType. Ex: utMass, utDistance, utMassPerArea, etc. This is further classified as
    - Simple unit type: Represented by single unit like *m, l, in, gal, lb, etc*
    - Composite unit type: A combination of multiple simple units with an integer power to indicate degree of the factor. Ex: m^2, *lb/ac, 1/hz,* cm^3, kg/m^2
- Each unit type contains a list of supported unit of measures. Each unit of measure has a scale, base offset, domain id (label) and a name which can be localized.
    - A base unit of measure is selected to be of scale 1 in the unit of measure list and every other unit is a conversion from that scale.
    - Some unit types can have both simple and composite unit type. Ex: area which can have simple units like *ac, ha* or composite units like m^2. The composite unit has a scale which is derived from the base unit in the simple unit in this case.
    - A base offset is used to offset a converted value with an additional value. Ex: temperature where the value needs to be multiplied as well as offset by a certain value.
    - The domain id is a unique value and is also used as the label. Ex: *m, l, gal, etc*.
- A unit type can also be a composite. The xml file contains the possible combination along with what is the power.
    - Ex: utMassPerArea indicates a unit type of Mass with a power 1 to indicate it is in the numerator and a unit type which is area with a power -1 to indicate it is the denominator.
    - A composite unit type can be a combination of multiple simple types or composite types. In the above example, area has both simple and composite unit type. It can be a simple unit as *ac* or a composite unit like m^2.
    - The composite unit type does not contain the actual combination of units but just lists the possible combination of unit types. Ex: utMassPerVolume, utDistancePerDegree.
    - When converting composite units, the scale is calculated by using the simple type defined in the numerators and denominator. Ex: for utMassPerArea, the scale is calculated by using the base unit for mass and area multiplied by the scale of the unit that it has to be converted from.
    - The domain id and label for composite units are generated dynamically using the power. Ex:
        - lb/ac is represented as lb1ac-1. lb is the numerator, 1 is the delimiter, ac is the denominator indicated by power -1.
        - lb/m^2 is represented as lb1[m2]-1. lb is the numerator, 1 is the delimiter, square brackets indicate degree. The square brackets contain a simple unit followed by

the degree. In this example it m2 indicates m^2. -1 after the square bracket indicates that it is the denominator.
        - The label is generated from the units represented in the domain id. If a unit lb1ac-1 is created, it creates a label lb/ac and a name Pounds per Acre using the name defined for the simple unit.
    - The composite unit conversion will work even for composite unit types not defined in the xml file.
        - Ex: a conversion can exists for $/ac to euro/ac. Currency is a valid unit type and area is a valid unit type. The system can create the scale from the numerator and denominator unit types.
        - This means that composite units need not be defined in the xml file. The only reason they are defined is to have them loaded into memory to avoid creating multiple units at run time when processing data.
- The units can be further classified into unit of measure systems which is umsEnglish, umsMetric and umsImperial. This system works in conjunction with the representation system (equivalent to DDI's in ISO world). This allows representation to have the base and preferred units defined for English, metric and imperial system. It can also classify simple and composite units by English, metric and imperial system to provide a grouping by the units that need to be supported.
    - Ex: English units for mass will contain lb, ton, ozm, troyoz, etc. The metric units for mass will contain g, kg, mg, etc.
    - Ex: Mass per area. English lb/ac; Metric kg/m^2; Imperial: lb/m^2

BaseNumber API

The codebase uses a base number to represent a value along with a unit. The base number can be used to create a number with a source unit of measure and convert it to a different unit by setting the target unit of measure.

The base numbers can be Added, Subtracted, Multiplied and Divided between

1) Simple and simple units. Ex:
    a. lb * kg
    b. lb * lb
    c. in / ft
2) Composite and composite unit: Ex
    a. lb/ac * kg/ha
    b. (kg/ha) / (kg/ha)
3) Simple and composite unit
    a. lb/ac * ac
    b. l/hour * sec
4) Complex composite units

a. (l/g) / (kg/g) => l/kg
b. g/ha * kg/g => kg/ha

XML example:

Simple Unit:

```
<UnitType domainID="utForce">
	<UnitTypeRepresentation domainID="urForce">
		<UnitOfMeasure scale="1" baseOffset="0" domainID="N">
			<Name label="N" locale="en" plural="N">Newton</Name>
			<Name label="N" locale="de" plural="N">Newton</Name>
			<Name label="N" locale="fr" plural="N">Newton</Name>
		</UnitOfMeasure>
		<UnitOfMeasure scale="1.00E-03" baseOffset="0" domainID="mN">
			<Name label="mN" locale="en" plural="mN">milliNewton</Name>
			<Name label="mN" locale="de" plural="mN">milliNewton</Name>
			<Name label="mN" locale="fr" plural="mN">milliNewton</Name>
		</UnitOfMeasure>
		<UnitOfMeasure scale="4.448222" baseOffset="0" domainID="lbf">
			<Name label="lbf" locale="en" plural="pounds of force">pounds of force</Name>
			<Name label="lbf" locale="de" plural="Pound-Force">Pound-Force</Name>
			<Name label="lbf" locale="fr" plural="livres-force">livres-force</Name>
		</UnitOfMeasure>
	</UnitTypeRepresentation>
	<Name locale="en">force</Name>
	<Name locale="de">Kraft</Name>
	<Name locale="fr">force</Name>
</UnitType>
```

Simple unit type containing both simple and composite representations:

```
<UnitType domainID="utArea">
	<UnitTypeRepresentation domainID="urArea">
		<UnitOfMeasure domainID="ha" scale="1" baseOffset="0">
			<Name locale="en" label="ha" plural="hectares">hectare</Name>
			<Name locale="de" label="ha" plural="Hektar">Hektar</Name>
			<Name locale="fr" label="ha" plural="hectares">hectare</Name>
		</UnitOfMeasure>
		<UnitOfMeasure domainID="ac" scale="0.40468564224" baseOffset="0">
			<Name locale="en" label="ac" plural="acres">acre</Name>
			<Name locale="de" label="ac" plural="Acres">Acre</Name>
			<Name locale="fr" label="ac" plural="acres">acre</Name>
		</UnitOfMeasure>
		<UnitOfMeasure domainID="thsndSqFt" scale="0.009290304" baseOffset="0">
			<Name locale="en" label="1000 sq.ft" plural="thousand square feet">thousand square feet</Name>
			<Name locale="de" label="1000 sq.ft" plural="Tausend Square Feet">Tausend Square Feet</Name>
			<Name locale="fr" label="1000 sq.ft" plural="milliers de square feet">milliers de square feet</Name>
		</UnitOfMeasure>
	</UnitTypeRepresentation>
	<CompositeUnitTypeRepresentation domainID="urAreaDistanceSquared" scale="0.0001" baseOffset="0">
		<UnitTypeRef unitTypeRef="utDistance" power="2" baseUnitOfMeasureRef="m"/>
	</CompositeUnitTypeRepresentation>
	<Name locale="en">area</Name>
	<Name locale="de">Fläche</Name>
	<Name locale="fr">surface</Name>
```

```
</UnitType>
```

Simple unit with base offset:

```
<UnitType domainID="utTemperature">
	<UnitTypeRepresentation domainID="urTemperature">
		<UnitOfMeasure domainID="C" scale="1" baseOffset="0">
			<Name locale="en" label="°C" plural="celsius">celsius</Name>
			<Name locale="de" label="°C" plural="Celsius">Celsius</Name>
			<Name locale="fr" label="°C" plural="Celsius">Celsius</Name>
		</UnitOfMeasure>
		<UnitOfMeasure domainID="F" scale="0.555556" baseOffset="-17.777777778">
			<Name locale="en" label="°F" plural="fahrenheit">fahrenheit</Name>
			<Name locale="de" label="°F" plural="Fahrenheit">Fahrenheit</Name>
			<Name locale="fr" label="°F" plural="Fahrenheit">Fahrenheit</Name>
		</UnitOfMeasure>
		<UnitOfMeasure domainID="K" scale="1" baseOffset="-273.15">
			<Name locale="en" label="K" plural="kelvin">kelvin</Name>
			<Name locale="de" label="K" plural="Kelvin">Kelvin</Name>
			<Name locale="fr" label="K" plural="kelvin">kelvin</Name>
		</UnitOfMeasure>
	</UnitTypeRepresentation>
	<Name locale="en">temperature</Name>
	<Name locale="de">Temperatur</Name>
	<Name locale="fr">température</Name>
</UnitType>
```

Composite unit:

```
<UnitType domainID="utMassPerArea">
	<CompositeUnitTypeRepresentation domainID="urMassPerArea" scale="1" baseOffset="0">
		<UnitTypeRef unitTypeRef="utMass" baseUnitOfMeasureRef="g"/>
		<UnitTypeRef unitTypeRef="utArea" baseUnitOfMeasureRef="ha" power="-1"/>
	</CompositeUnitTypeRepresentation>
	<Name locale="en">mass per area</Name>
	<Name locale="de">Masse je Fläche</Name>
	<Name locale="fr">masse/zone</Name>
</UnitType>
```

Code examples:

1) Loading simple unit: `UnitSystemManager.Instance.UnitOfMeasures[“lb”];`
2) Loading composite unit: `UnitSystemManager.Instance.UnitOfMeasures[“lb1ac-1”];`
3) Converting simple units:

```
UnitOfMeasure uom1 = UnitSystemManager.Instance.UnitOfMeasures[“ac”];
UnitOfMeasure uom2 = UnitSystemManager.Instance.UnitOfMeasures[“m2”];

BaseNumber number = new BaseNumber(10, uom1);
number.TargetUnitOfMeasure = uom2;

Console.WriteLine(“{0} {1} = {2} {3}”, number.SourceValue,
number.SourceUnitOfMeasure.Label, number.TargetValue,
number.TargetUnitOfMeasure.Label);
```

//output
// 10 ac = 40468.5642 m2

4) Converting composite units:

```
UnitOfMeasure uom1 = UnitSystemManager.Instance.UnitOfMeasures[“gal1ac-1”];
UnitOfMeasure uom2 = UnitSystemManager.Instance.UnitOfMeasures[“l1[m2]-1”];

BaseNumber number = new BaseNumber(1000, uom1);
number.TargetUnitOfMeasure = uom2;

Console.WriteLine(“{0} {1} = {2} {3}”, number.SourceValue,
number.SourceUnitOfMeasure.Label, number.TargetValue,
number.TargetUnitOfMeasure.Label);
```

//output
// 1000 gal/ac = 0.935395623 l/m2

```
Console.WriteLine(“{0} {1} = {2} {3}”, number.SourceValue,
number.SourceUnitOfMeasure.Name, number.TargetValue,
number.TargetUnitOfMeasure.Name);
```

//output
// 1000 Gallons per Acre = 0.935395623 Liters per Square Meter

5) Multiplying simple with composite unit.

```
UnitOfMeasure lPerHaUom = UnitSystemManager.Instance.UnitOfMeasures["l1hr-1"];
UnitOfMeasure secUom = UnitSystemManager.Instance.UnitOfMeasures["sec"];

BaseNumber lPerHaBaseNumber = new BaseNumber(0.215, lPerHaUom);
BaseNumber secBaseNumber = new BaseNumber(1, secUom);

BaseNumber result = BaseNumber.Multiply(lPerHaBaseNumber, secBaseNumber);
Assert.AreEqual(0.215 / 3600.0, result.TargetValue);
Assert.AreEqual(“l”, result.TargetUnitOfMeasure.DomainId);
Assert.AreEqual(“l”, result.TargetUnitOfMeasure.Label);
```

6) Dividing simple units

```
var m3Uom = UnitSystemManager.Instance.UnitOfMeasures["m3"];
var acUom = UnitSystemManager.Instance.UnitOfMeasures["ac"];

var expectedMetersCubedPerAcre = new BaseNumber(8,
UnitSystemManager.Instance.UnitOfMeasures["[m3]1ac-1"]);

var expectedCentimetersCubedPerAcre = new BaseNumber(8000000,
UnitSystemManager.Instance.UnitOfMeasures["[cm3]1ac-1"]);

var volume = new BaseNumber(16, m3Uom);
var area = new BaseNumber(2, acUom);
```

```
Assert.IsTrue(expectedMetersCubedPerAcre.IsEqual(volume / area));

Assert.IsTrue(expectedMetersCubedPerAcre.IsEqual(BaseNumber.Divide(volume,
area)));

Assert.IsTrue(expectedMetersCubedPerAcre.IsEqual(BaseNumber.Divide(volume, area,
false)));

Assert.IsTrue(expectedCentimetersCubedPerAcre.IsEqual(BaseNumber.Divide(volume,
area, false, UnitSystemManager.Instance.UnitOfMeasures["[cm3]1ac-1"])));
```