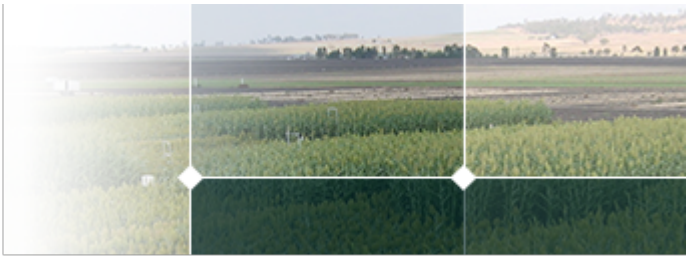


1 Manager Model Documentation and Examples

Manager is the model in APSIM that is used to script the on-farm management of a simulation. The examples in this document increase in complexity. Read this to better understand the structure of a manager script

2 NTopupSimulation

2.1 Change sowing fertiliser to a topup rule

This simulation started with the sowing fertiliser rule from the management toolbox.

```
using System;
using Models.Core;
using Models.PMF;
namespace Models
{
	[Serializable]
	public class Script : Model
	{
		[Link] Clock Clock;
		[Link] Fertiliser Fertiliser;

		[Description("Crop to be fertilised")]
		public IPlant Crop { get; set; }

		[Description("Type of fertiliser to apply? ")]
		public Fertiliser.Types FertiliserType { get; set; }

		[Description("Amount of fertiliser to be applied (kg/ha)")]
		public double Amount { get; set; }

		[EventSubscribe("Sowing")]
		private void OnSowing(object sender, EventArgs e)
		{
			Model crop = sender as Model;
			if (Crop != null && crop.Name.ToLower() == (Crop as
IModel).Name.ToLower())
				Fertiliser.Apply(Amount: Amount, Type: FertiliserType);
		}
	}
}
```

This script has 3 user-input properties, the sown crop model that the rule applies to, the fertiliser type to apply and the amount of fertiliser to apply:

```
[Description("Crop to be fertilised")]
public IPlant Crop { get; set; }

[Description("Type of fertiliser to apply? ")]
public Fertiliser.Types FertiliserType { get; set; }

[Description("Amount of fertiliser to be applied (kg/ha)")]
public double Amount { get; set; }
```

For a simple N topup rule we don't need the crop or the fertiliser type and instead of specifying an amount to apply, we'll change this to a threshold amount to which the script will keep the NO3 level to. We'll rename this property to *NThreshold* :

```
      [Description("Threshold amount of NO3 below which fertiliser will be added
(kg/ha)")]
      public double NThreshold { get; set;}
```

Next we need to add a link to the NO3 solute as we'll need this to determine how much to apply. The new Link will look like this:

```
[Link(ByName=true)] Solute NO3;
```

This link looks different to the others because there are multiple solutes in the simulation and we want to make sure we are getting the NO3 solute. A regular *[Link]* looks for a model of the correct type and doesn't use the name to perform the match. If we used a regular link for this solute then we will get a link to the first solute found which may not be NO3. To fix this we need to specify that ByName=true, meaning that the link will be resolved by looking at the name of the variable (NO3 in this case) to do the match.

For this link to work though, we need to include another namespace. The nutrient model and all solutes are contained in the *Models.Soils.Nutrients* namespace so we need to add this to the top of the script:

```
      using Models.Soils.Nutrients;
```

Currently in the script there is a OnSowing method that gets called whenever a sowing event occurs. We need to change this to a method that gets called every day of the simulation. Most management scripts use DoManagement for this purpose.

```
      [EventSubscribe("DoManagement")]
      private void OnDoManagement(object sender, EventArgs e)
```

The event name is *DoManagement* and this goes in the* EventSubscribe* attribute. The convention then is to name the method the same as the event but with *On* prefixed to the front of the event name.

Now we can perform the calculation of the amount of NO3 to apply.

```
      double NAmount = Threshold - MathUtilities.Sum(NO3.kgha);
```

NO3.kgha is an array variable (double[[]]) with one value per layer. We need to sum this over the whole array so we use a *MathUtilities.Sum* function to do this. For this to compile though we need to include another namespace at the top of the script

```
      using APSIM.Shared.Utilities;
```

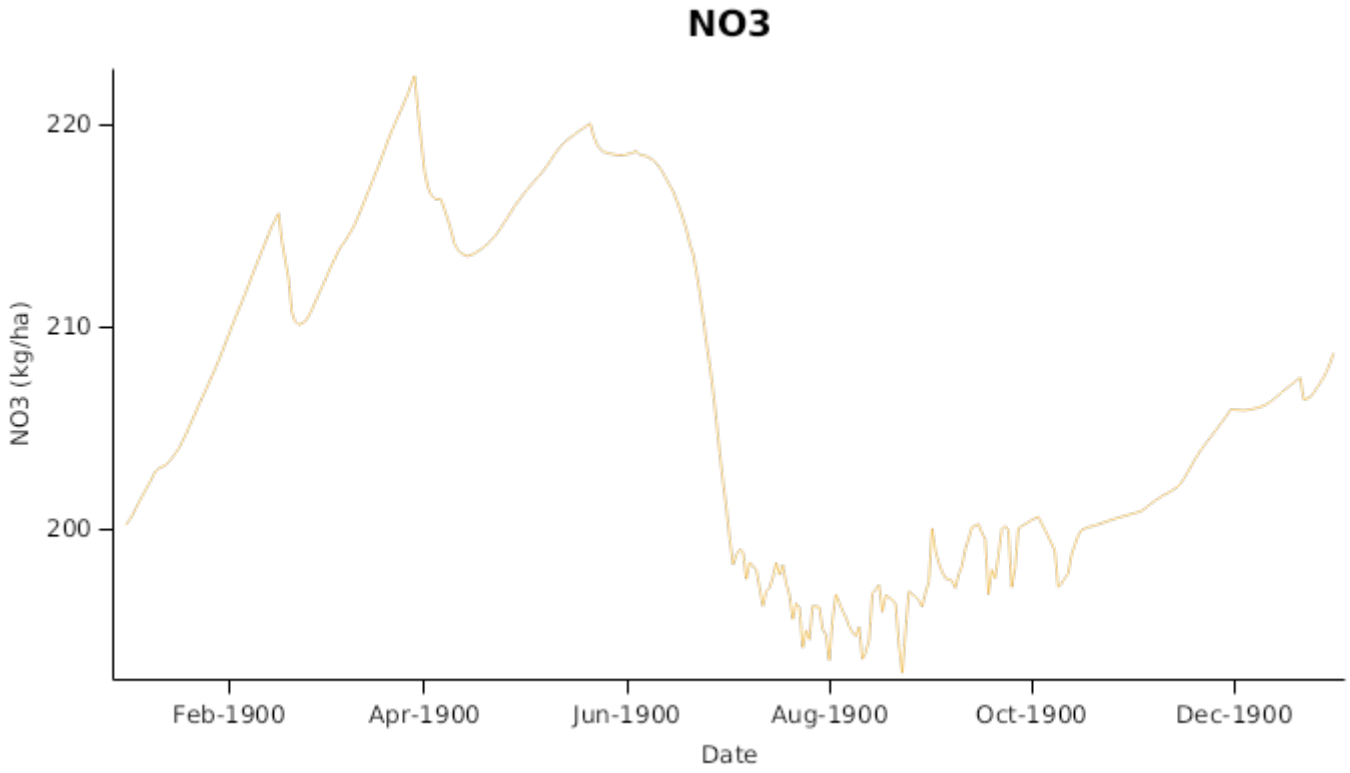
We can remove the existing references to *Crop* as this rule will apply all year around, not just within crop. We also hard code the fertiliser type on the apply line.

```
      Fertiliser.Apply(Amount: NAmount, Type: Fertiliser.Types.NO3N);
```

Finally, we rename the manager model to *NTopup*. The final script looks like:

```
      using System;
      using Models.Core;
      using Models.PMF;
      using Models.Soils.Nutrients;
      using APSIM.Shared.Utilities;
      namespace Models
```

```
      {
            [Serializable]
            public class Script : Model
            {
                  [Link] Clock Clock;
                  [Link] Fertiliser Fertiliser;
                  [Link(ByName=true)] Solute NO3;

                  [Description("Threshold amount of NO3 below which fertiliser will be
added (kg/ha)")]
                  public double NThreshold { get; set;}


                  [EventSubscribe("DoManagement")]
                  private void OnDoManagement(object sender, EventArgs e)
                  {
                        double NAmount = NThreshold - MathUtilities.Sum(NO3.kgha);
                        Fertiliser.Apply(Amount: NAmount, Type: Fertiliser.Types.NO3N);
                  }
            }
      }
```



3 WaterAndNTopupSimulation

3.1 Change N topup rule to N and water topup rule

This simulation extends the previous simulation by adding a water topup rule to the topup rule. This is useful when a potential crop/pasture yield is required.

This script has a single user-input property, the N threshold below which fertiliser will be applied.

```
      [Description("Threshold amount of NO3 below which fertiliser will be added
(kg/ha)")]
      public double NThreshold { get; set;}
```

This is still needed but we need to add a similar property for a water threshold.

```
      [Description("Threshold amount of water below which irrigation will be added
(mm)")]
      public double WaterThreshold { get; set;}
```

Next we need to add a link to the Soil model as we'll need this to get the current value of soil water.

```
      [Link] Soil Soil;
```

We then need to add an irrigation model to our field. Then we can add a link to it.

```
      [Link] Irrigation Irrigation;
```

For this link to work though, we need to include another namespace. The soil model is contained in the *Models.Soils* namespace so we need to add this to the top of the script:

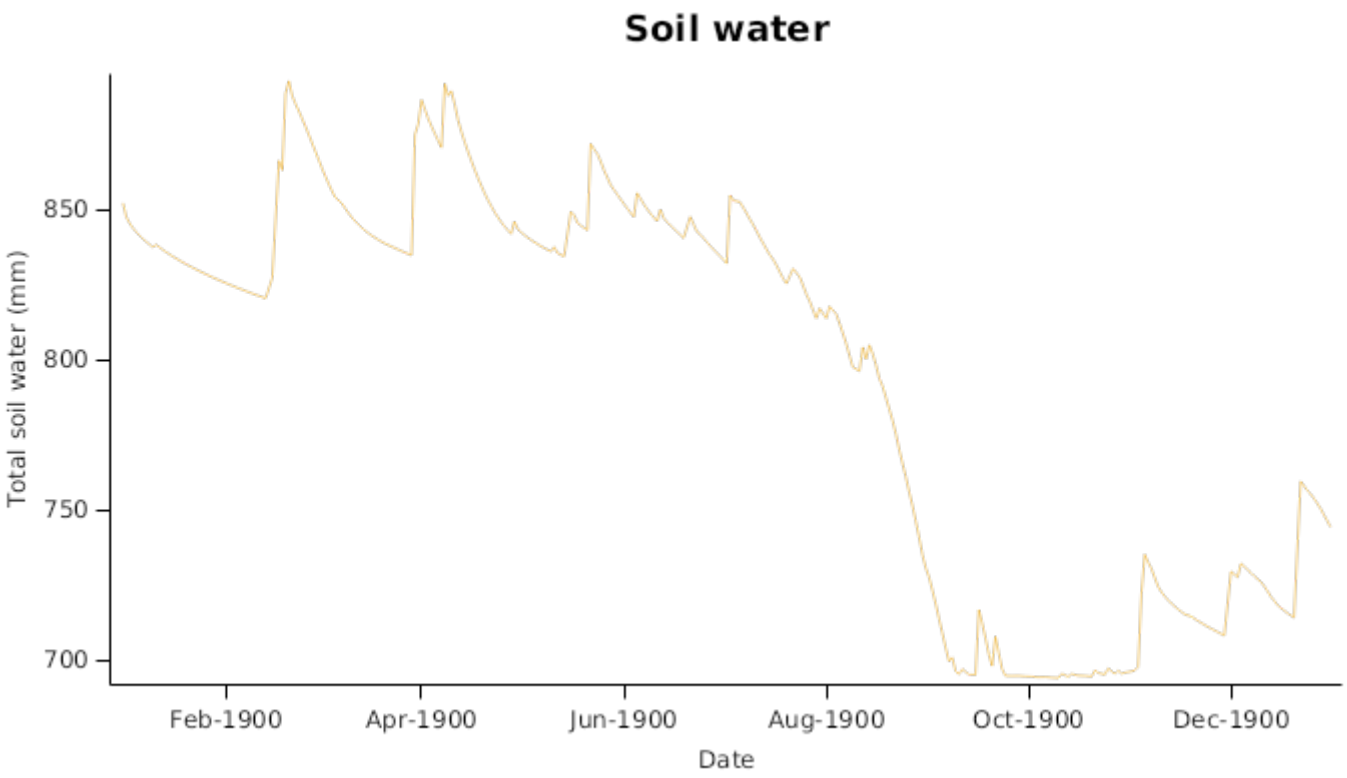
```
      using Models.Soils;
```

In the *OnDoManagement* script, we can calculate the amount of water to add then then apply the irrigation if the amount is greater than zero. double SWAmount = WaterThreshold - MathUtilities.Sum(Soil.Water); if (SWAmount > 0) Irrigation.Apply(SWAmount);

Finally, the manager model should be renamed to something more meaningful e.g. *NAndWaterTopup*. The final script looks like:

```
      using System;
      using Models.Core;
      using Models.PMF;
      using Models.Soils.Nutrients;
      using APSIM.Shared.Utilities;
      using Models.Soils;
      namespace Models
      {
            [Serializable]
            public class Script : Model
            {
                  [Link] Clock Clock;
                  [Link] Fertiliser Fertiliser;
                  [Link(ByName=true)] Solute NO3;
                  [Link] Soil Soil;
                  [Link] Irrigation Irrigation;

                  [Description("Threshold amount of NO3 below which fertiliser will be
added (kg/ha)")]
                  public double NThreshold { get; set;}

                  [Description("Threshold amount of water below which irrigation will be
added (mm)")]
                  public double WaterThreshold { get; set;}


                  [EventSubscribe("DoManagement")]
                  private void OnDoManagement(object sender, EventArgs e)
                  {
                        double NAmount = NThreshold - MathUtilities.Sum(NO3.kgha);
                        Fertiliser.Apply(Amount: NAmount, Type: Fertiliser.Types.NO3N);

                        double SWAmount = WaterThreshold - MathUtilities.Sum(Soil.Water);
                        if (SWAmount > 0)
                              Irrigation.Apply(SWAmount);
                  }
```

```
        }
    }
```


Soil water
Total soil water (mm)
850
800
750
700
Feb-1900
Apr-1900
Jun-1900
Aug-1900
Oct-1900
Dec-1900
Date