Tutorial II: Introduction to Envision – Part II

April 12, 2013

This tutorial extends the work done earlier with the Blue River watershed dataset. Blue River is a tributary of the McKenzie River, and we will move up to that larger river basin for this work. The primary goal for the exercise is to get additional experience in working with Envision, with a focus on adding plugin models (in this case an autonomous process) and further exploration of the policy framework as a mechanism to capture potential changes associated with human-decisionmaking.

1. Lesson #2: Getting Started

A. The McKenzie River study area

Step 1: Installing Envision

If Envision has not already been installed on your computer, the first step is to get the installation package from the website and run it. The best option is to install the software in a directory called [d:]\Envision. The drive letter (C, D, etc) does not matter. The 64-bit version of the software can be found at: http://envision.bioe.orst.edu/Downloads/Setup-x64.msi . A 32-bit version of the software can be found at: http://envision.bioe.orst.edu/Downloads/Setup.msi.

If you are running a 64-bit version of Windows, you should use the 64-bit version of Envision. Otherwise, install the 32-bit version. This installs the main Envision executable files and a standard set of plug-ins.

The next step is to get an example Envision study area project from the Envision website.

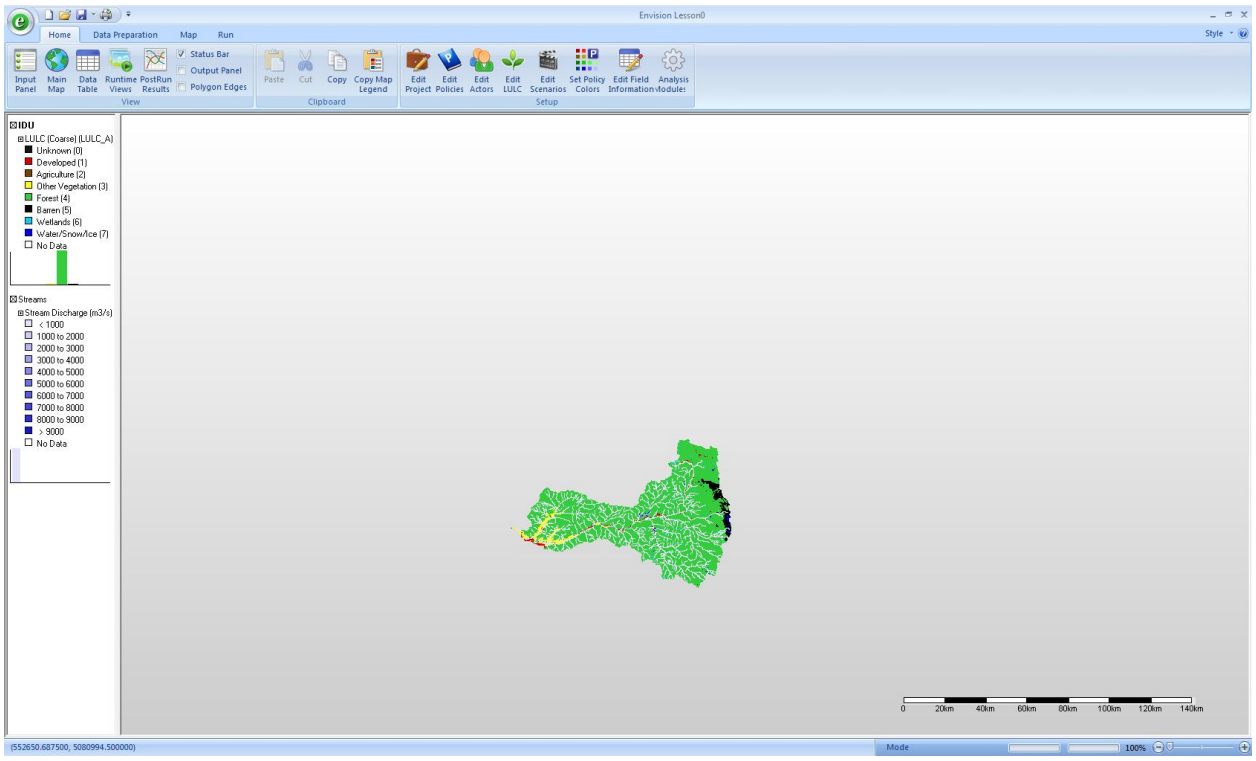
The datasets used in this tutorial are at http://envision.bioe.orst.edu/StudyAreas/Tutorials/Tutorials.msi. Using the default install directory will allow the files to run unmodified. When you do that, you should end up with a directory [d:]\Envision\StudyAreas\Tutorials\ that includes a couple of subdirectories and a number of different files.

At this point, you should have installed both the Envision executable files and a set of input files (termed a "Project" in Envision terminology, and are ready to run Envision.

1. Run Envision and you will get a popup widow like:

Click on “Find Existing Project” and map to the ‘Tutorials’ directory that contains the sample project. There are 3 different .envx files in the directory. For now, choose the file ‘Lesson2.envx’. As in previous work, if your directory is different the suggested structure, you will need to edit a number of paths in the file ’WW2100.envx’ for the project to load properly. In this case, replace all paths that you find in those ‘WW2100.envx’ with the correct directories for your particular machine.

When you open the project, there will be two map layers displayed. One is the polygon coverage of Integrated Decision Units (IDUs) and the other is the Stream network coverage.



The project includes two shapefiles, one called IDU and the other called Streams. The map is classified based on a particular attribute in the IDU shapefile, called LULC_A. You can zoom in a bit closer to the map using the tools under Map. At this point, you can run the model and see the results of the DynamicVeg plugin that we explored in Lesson 1, but played out across a larger basin. Note that the simulation length is now 100 years, rather than the 300 we looked at earlier.

2. Lesson #2: Population Growth and Development

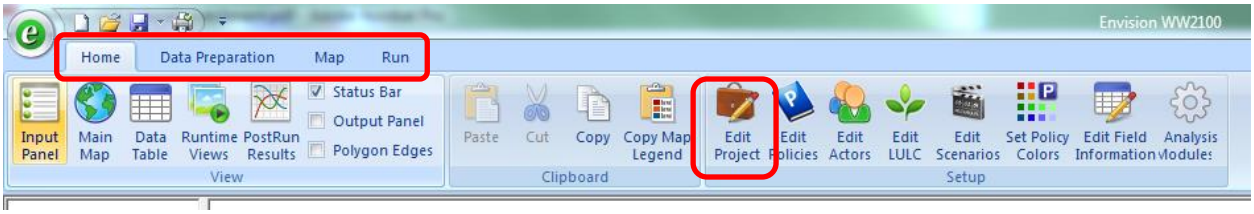
Next, we will demonstrate the process of adding a plug-in model to this Envision application. The model will simulate the process of representing human population growth in the study area, and will discuss the problem of spatially allocating that population growth. To do this, we will use one of Envision's "standard" plug-ins, referred to as "Target". We will go through the process of adding a plug-in and configuring that plug-in for the application.

A. Adding a plugin model to Envision

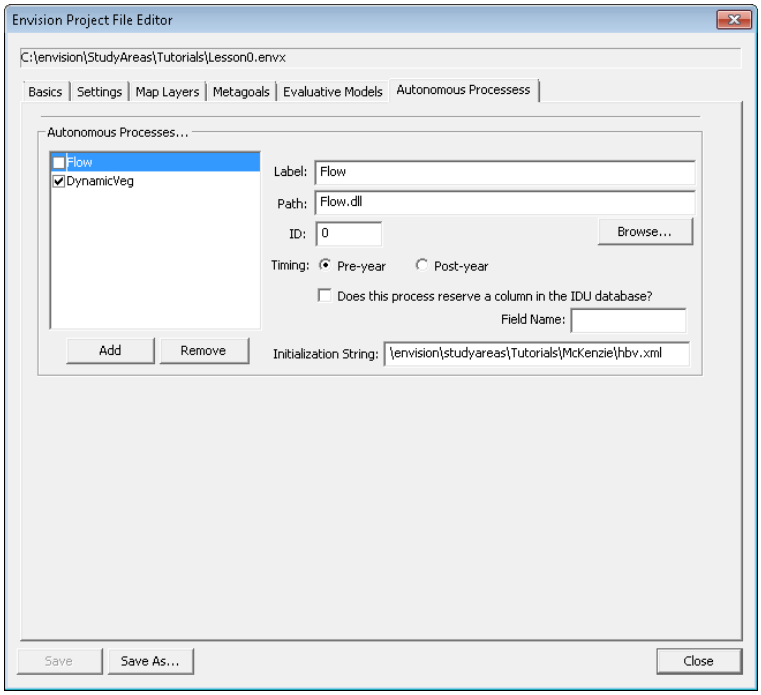
Here will we introduce the inclusion of a plug-in model that allows the human population to grow at some overall study wide growth rate, and spatially allocates that growing population onto the IDU landscape representation. In this example, the allocation of growth involves an IDU by IDU evaluation of existing population and of the capacity for new population, which here we characterize through existing zoning. A key assumption is that the growing population will be allocated to the landscape in proportion to the available capacity for new growth. An undeveloped lot in an urban area (say it is zoned to allow 6 houses/acre) is more likely to see new population as compared to an undeveloped agricultural field (eg. 1 house/40 acres). We use an existing Envision plug-in (Target); the code for the process exists as a dynamic link library (DLL).

Population Growth – A Conceptual Model

For this example, we assume that population growth occurs at some known, fixed rate for the study area. We further assume that the landscape has controls on where new population can go; specifically, we assume that zoning exists that regulates growth allocation. These controls generate a **capacity surface**, a spatial representation of the maximum density of people that can legally exist in a given IDU. Along with this, we assume there is an **existing density surface** that represents the density of people at any given point in time for each IDU. The conceptual model for allocating new growth assumes that the existing density surface "grows" towards the capacity surface, with new growth allocated proportionally according to the "distance" between the capacity surface and the existing density surface. We further allow that this "difference surface" can be warped to reflect preferences for new growth to be allocated to locations that have particular qualities, e.g. within urban areas, or areas near amenities. This conceptual model is implemented using Envisions *Target* plug-in.



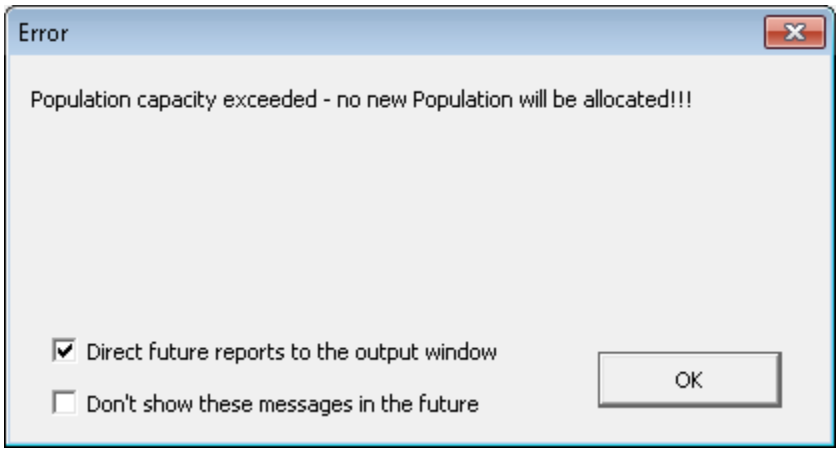
You will get the Project Editor Dialog, and will switch to the Autonomous Processes tab:



Clicking <Add> will bring up a dialog box listing all the Plug-in DLLs in the Envision directory. If your plug-in is there, select it; otherwise, navigate to the location where your plug-in is located. For this exercise, we want to load the "Target.dll" plug-in, so select Target.dll in the Open File dialog box. You can then change the default label to something like 'Population' and, optionally, you can delete the directory portion of full path from the Path dialog box (Envision can generally find these DLLs without the full path specified if they are located in the Envision executable directory. You then will need to add a text string to the Initialization String dialog box. This string points Target to a file that it will read as input data and should be "\envision\StudyAreas\Tutorials\McKenzie\TargetPopulation.xml" (leave off the quotes!) If you put the project files in a different directory, modify the string to match. The input is stored in an Xml format (see http://www.w3schools.com/xml/xml_whatis.asp if you are interested in some additional detail about what Xml is), which we will encounter in a number of different Envision exercises. The file can be read in any text editor. The last step is to check the box next to Target in the listing of available models on the left side of the dialog box. Then close the dialog, and save the changes.

NOTE: this particular change will not be included in the project until you exit Envision and reload.

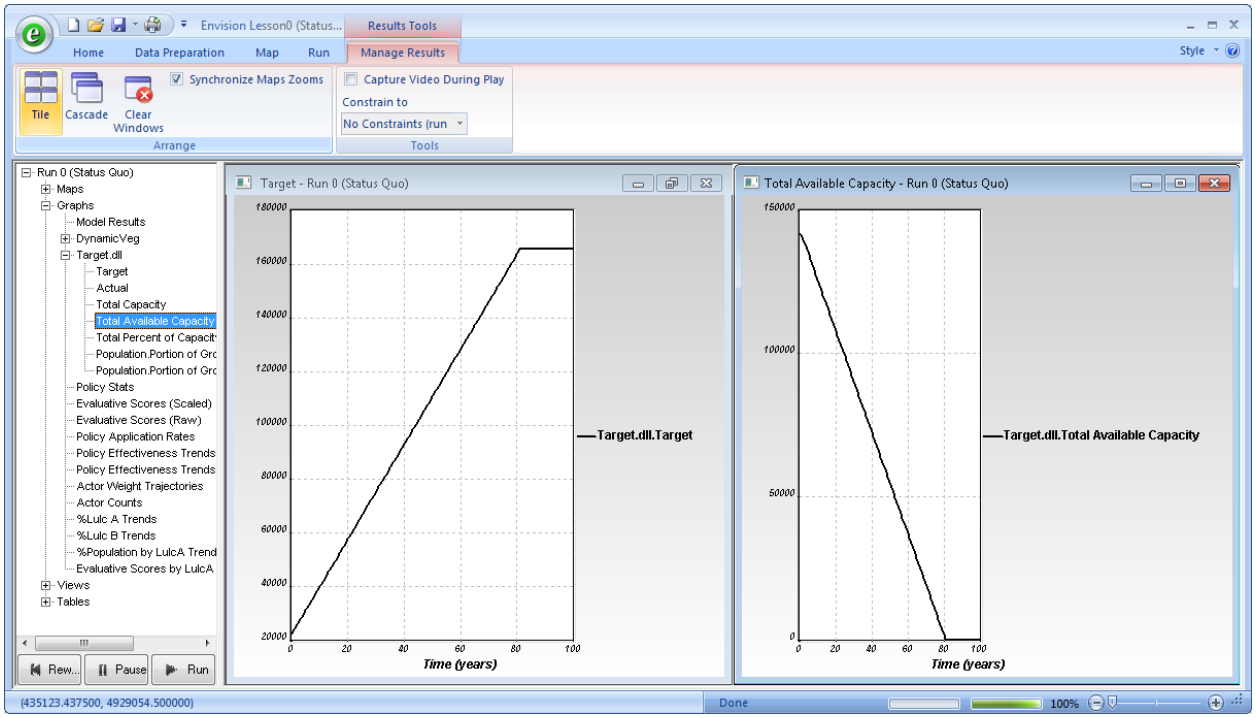
Reload Envision, and run the scenario. Population will now grow (at 8%/year) and will be allocated to IDUs based on available capacity for new growth. At this point, the model is not changing zoning and given 8% growth over many years, the landscape may run out of capacity for new growth. Envision will report that with a dialog box like:



Click 'Don't show these messages in the future' and OK to keep the same dialog from popping up in each of the remaining years.

Now take a look at some results. Click to PostRunResults and open the Graphs->Target section of the hierarchy. Double-click Target and then Total Available Capacity to display two time series graphs outlining the overall population in the study region and also the available capacity for growth, similar to those shown below. The total capacity is characterized by zoning data, and runs out when the population gets to about 161,000 people. In this case, that occurs in 2086, and resulted in the dialog above.

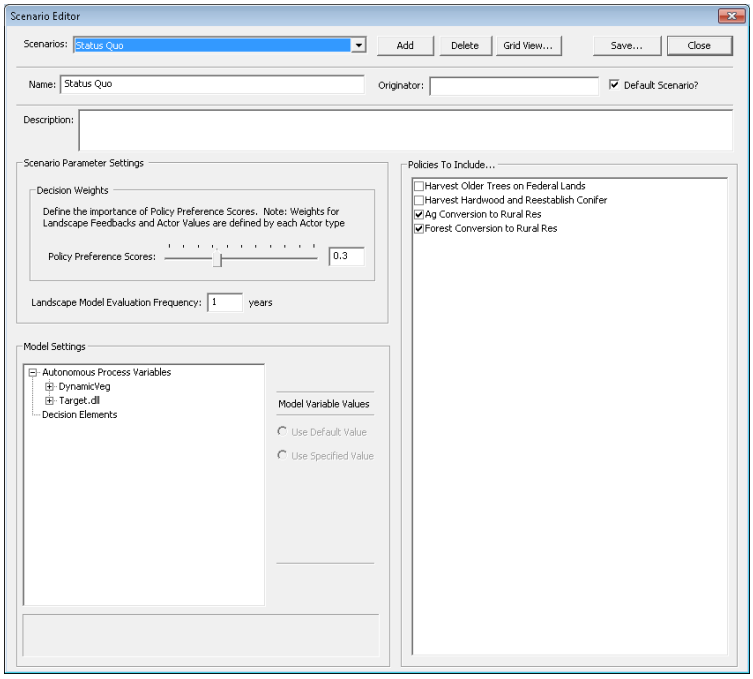
Feel free to take a look at some of the other Graphs and Maps.



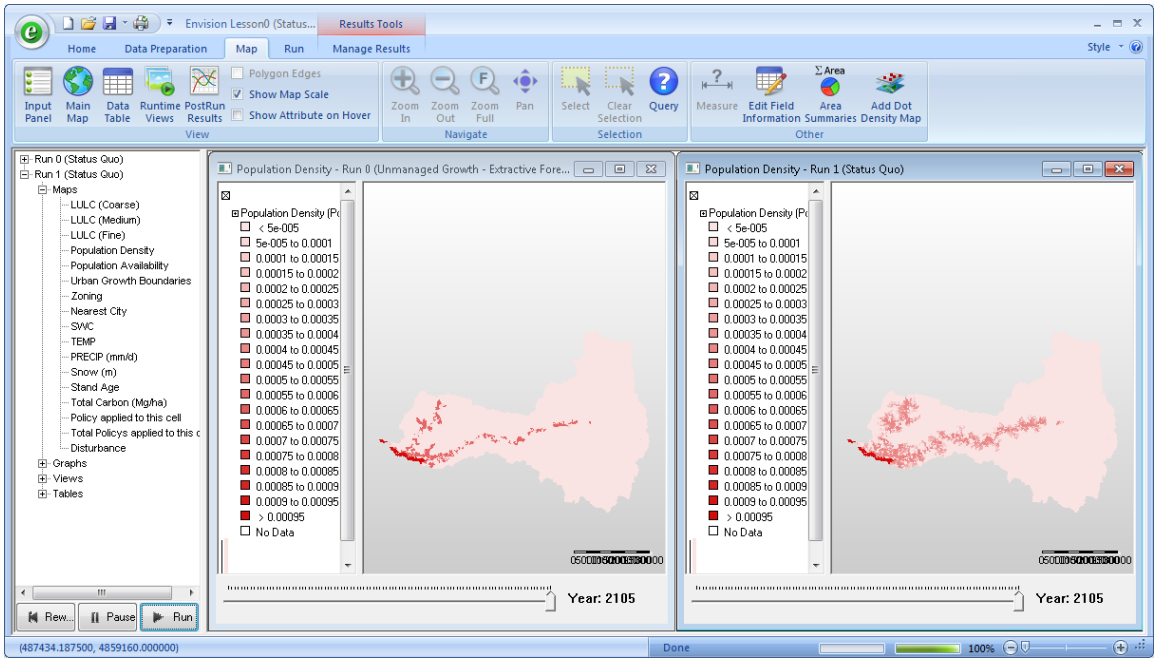
3. Expanding capacity for new development through the policy framework.

Existing zoning regulations are generally designed for anywhere from 20-30 years and adjustments are expected along those timeframes. Given that, it is not surprising that we run completely out of capacity after 80 or so years. In this section, we will use policies to expand capacity and redirect growth to different areas of the landscape.

Two new policies (beyond those we explored in an earlier exercise) have been included in the McKenzie project. You will see them listed in the scenario editor. It should look something like this:



If you turn on the Ag and/or Forest Conversion policies for one of the scenarios, save the changes, and then rerun the model you will have a second set of results that include the influence of these newly included rezoning strategies. You could then explore comparisons between the different model runs. If you compare maps of population density you might see something like the following representing the final year of the simulations:



Explore some of the other results and then return to the policy framework. Develop some different policies to explore how modifications to zoning can be included and how this might result in a different pattern of population growth in the study region over the next 100 years. Be creative and don't be constrained by reality – at this point, this is entirely exploratory.

Appendix 1: Some additional details about Target

The population growth and development model consists of an initial map, at the IDU scale, capturing population density. This initial map is often developed to be consistent with the census block data for the region. The IDU map may include the block geometry as part of its definition, and in that case, there are always multiple IDUs within each census block. A downscaling procedure is used to attribute individual IDUs with population density, insuring that the sum of population across any census block is consistent with the original data.

Population growth and development is modeled as a two step process involving interplay between policies setting up development opportunities and an autonomous process, *Target*, that allocates new population. *Target* works by creating and examining two surfaces: 1) a current population density surface, and 2) a population capacity surface. The population capacity surface represents the potential of the IDU's to contain population density. It is defined in spatially-dependent and scenario-specific terms in the input file to the target.dll process. Total population targets were computed for this project assuming a 8.0% yearly overall growth rates. This value can be changed by modifying the TargetPopulation.xml input file

Population capacity estimates are computed dynamically during scenario runs and will reflect any changes to zoning that might be associated with implemented policies. These population capacity surfaces are derived from estimates of maximum population density achieved within a zoning class at build out. Current population capacity surfaces were generated by relating existing population density surfaces to a set of a relevant underlying landscape attributes, expressed as a spatial query. An available population capacity (APC) is generated as the difference, at any given time and place, between the total population capacity and the current population capacity. To allocate new population as a scenario unfolds temporally, an allocation of growth is determined based the overall growth rate, which is then disaggregated down to the IDU level based on the APC at the IDU relative to the APC expressed across the entire analysis area. Thus, locations with larger APC received a proportionally larger share of the total allocation compared to locations with a smaller APC.

The following is a copy of the file called TargetPopulation.xml Text in Green represent comments that are ignored by the model, but useful in terms of describing different elements of the file. For the most part, brown or red text represent xml flags that the Envision code interprets. Blue text represent values that Envision will read and use for each of the red flags.

```
<?xml version="1.0" encoding="utf-8"?>

<!--
Target specifies a landscape variable whose total value across the landscape  is represented as a
trajectory, described as a rate or as a time series, and a set of "allocations" that describe a target
distribution surface in terms of underlying spatial attributes.

An "allocation" consists of a spatial query defining the landscape region where the allocation is
targeted, and a target equation that describes, in dimensionless, relative terms, the portion of the total
target allocated to that region. Attributes of an allocation include:
  name:       a string describing the allocation (e.g. "population center")
  query:      a spatial query (using Envision spatial query syntax) define where this allocation
              is located in teh underlying IDU coverage
  value:      a mathematical expression (supporting IDU field references) that returns a relative
              value of the future target surface.
  multiplier: (optional) a floating point value that scales up (>1) or down (0-1) the value expression

Note that if allocation queries (regions) overlap, the one specified FIRST has precedence over successive
allocations.

Multiple allocation scenarios can be described for a given target by specifying multiple <allocations>
blocks, each with a unique "id" (integer) and "name" attribute.

NOTE:  capCol, availCol contains absolute numbers, not per area numbers
       pctAvail = decimal percent
       computed "value" is per unit area.  It is used to populate capCol, but is put on an absolute basis
first (capCol = value * AREA )
-->

<target_process>

  <target name="Population" method="rateLinear" value="0.08" col="POPDENS"  capCol='POP_CAP'
availCol='POP_AVAIL' pctAvailCol='P_POP_AVAI' >
  <!--- target method="timeseries" value="(0,40000),(10,80000)" -->

  <report id='1' name='Portion of Growth Added Within UGB' query='UGB &gt; 0' />
  <report id='2' name='Portion of Growth Added Outside UGB' query='UGB = 0' />

  <const name="peoplePerDU" value="2.3" />

  <!-- the following are in DU/acre -->
  <const name="densUrban" value="4" />
  <const name="densRR" value="1" />
  <const name="densRCS" value="0" />
  <const name="densRC" value="0.005" />
  <const name='densAg' value='0.025' />
  <const name='densFor' value='0.020' />

  <!-- density multipliers for each scenario -->
  <const name='kSQ' value ='1.0' />
  <const name='kCG' value ='2.0' />
  <const name='kUG' value ='0.5' />

  <!--
  Status Quo Scenario:  Capacity stays similar to present
  -->

  <allocation_set id="0" name="Status Quo">
    <allocation name="Urban"              query='zone=15'  value="kSQ*densUrban*peoplePerDU/4047"
multiplier="1.0" />
```

```
    <allocation name="Agriculture"        query='zone=2'   value="densAg*peoplePerDU/4047"
multiplier="1.0" />
    <allocation name="Forestry"           query='zone=4'   value="densFor*peoplePerDU/4047"
multiplier="1.0" />
    <allocation name="Ag/Forestry"        query='zone=1'   value="(densFor+densAg)*peoplePerDU/(2*4047)"
multiplier="1.0" />
    <allocation name="RuralResidential"   query='zone=13'  value="densRR*peoplePerDU/4047"
multiplier="1.0" />
    <allocation name="RuralServiceCenter" query='zone=14'  value="0"
multiplier="1.0" />
    <allocation name="RuralIndustry"      query='zone=12'  value="densRC*peoplePerDU/4047"
multiplier="1.0" />
    <allocation name="Non-populated"      query='zone!=15 and zone !=2 and zone != 4 and zone != 1 and
zone != 13 and zone != 12 and zone != 14' value="0"  multiplier="1.0" />
  </allocation_set>

  <allocation_set id="1" name="Compact Growth">
    <allocation name="Urban"              query='zone=15'  value="kCG*densUrban*peoplePerDU/4047"
multiplier="1.0" />
    <allocation name="Agriculture"        query='zone=2'   value="densAg*peoplePerDU/4047"
multiplier="1.0" />
    <allocation name="Forestry"           query='zone=4'   value="densFor*peoplePerDU/4047"
multiplier="1.0" />
    <allocation name="Ag/Forestry"        query='zone=1'   value="(densFor+densAg)*peoplePerDU/(2*4047)"
multiplier="1.0" />
    <allocation name="RuralResidential"   query='zone=13'  value="densRR*peoplePerDU/4047"
multiplier="1.0" />
    <allocation name="RuralServiceCenter" query='zone=14'  value="0"
multiplier="1.0" />
    <allocation name="RuralIndustry"      query='zone=12'  value="densRC*peoplePerDU/4047"
multiplier="1.0" />
    <allocation name="Non-populated"      query='zone!=15 and zone !=2 and zone != 4 and zone != 1 and
zone != 13 and zone != 12 and zone != 14' value="0"  multiplier="1.0" />
  </allocation_set>

  <allocation_set id="2" name="Unmanaged Growth">
    <allocation name="Urban"              query='zone=15'  value="kUG*densUrban*peoplePerDU/4047"
multiplier="1.0" />
    <allocation name="Agriculture"        query='zone=2'   value="densAg*peoplePerDU/4047"
multiplier="1.0" />
    <allocation name="Forestry"           query='zone=4'   value="densFor*peoplePerDU/4047"
multiplier="1.0" />
    <allocation name="Ag/Forestry"        query='zone=1'   value="(densFor+densAg)*peoplePerDU/(2*4047)"
multiplier="1.0" />
    <allocation name="RuralResidential"   query='zone=13'  value="densRR*peoplePerDU/4047"
multiplier="1.0" />
    <allocation name="RuralServiceCenter" query='zone=14'  value="0"
multiplier="1.0" />
    <allocation name="RuralIndustry"      query='zone=12'  value="densRC*peoplePerDU/4047"
multiplier="1.0" />
    <allocation name="Non-populated"      query='zone!=15 and zone !=2 and zone != 4 and zone != 1 and
zone != 13 and zone != 12 and zone != 14' value="0"  multiplier="1.0" />
  </allocation_set>

</target>

</target_process>
```