Todos – updated 6/17/2016

* Restore support for dynamically generated models.
  + Most of the code should be there, just needs to be updated and tested
* Implement GUI stuff for Linux builds
  + Need to settle on a UI library to use (candidates: QT, Wx, TCL/TK,???)
* Look to incorporate more parallelization in non-update sections of code.
* Add additional views to Result browser.
  + The interface should be straightforward, and is designed to be modular
  + Here are the general steps for adding a new visualizer:
    - Create a representation of the data/options stored in the visualizer as a subclass of BaseResultsDetails in OMEDraw lib.
    - Create a control component in the interface wizard that will draw the output.
    - Create a control component in the interface wizard that will store controls for various view options, with at least a field used for naming the view.
    - Add a type in the ResultsMgr that will create an instance of the new visualizer when requested.
    - Add visualizer option to drop down in main window, and add logic that requests the view from the local ResultsMgr.
* Add a canvas view for constructing models natively, rather than relying on outside models.
* Expand support of Vensim features.
* Fully comply with model structure stuff with XMILE
* Incorporate compiler into model class builder. This is probably most easily accomplished using the Clang libraries from the LLVM project.
* Add python and/or java bindings to OMERuntime. Keep in mind that SimManager should be considered the common interface for interacting with the overall simulation.
* Extend unit testing stuff.
* Ensure Doxygen template exists for both MacOS, Linux.
* Extend the Linux cmake setup to also generate Visual Studio and XCode project files.