



(<https://www.eclipse.org/>)

Eclipse Newsletter (/community/eclipse_newsletter/) > January 2015 (/community/eclipse_newsletter/2015/january) > Upgrading Modeling and Simulation with the Eclipse Integrated Computational Environment

Upgrading Modeling and Simulation with the Eclipse Integrated Computational Environment

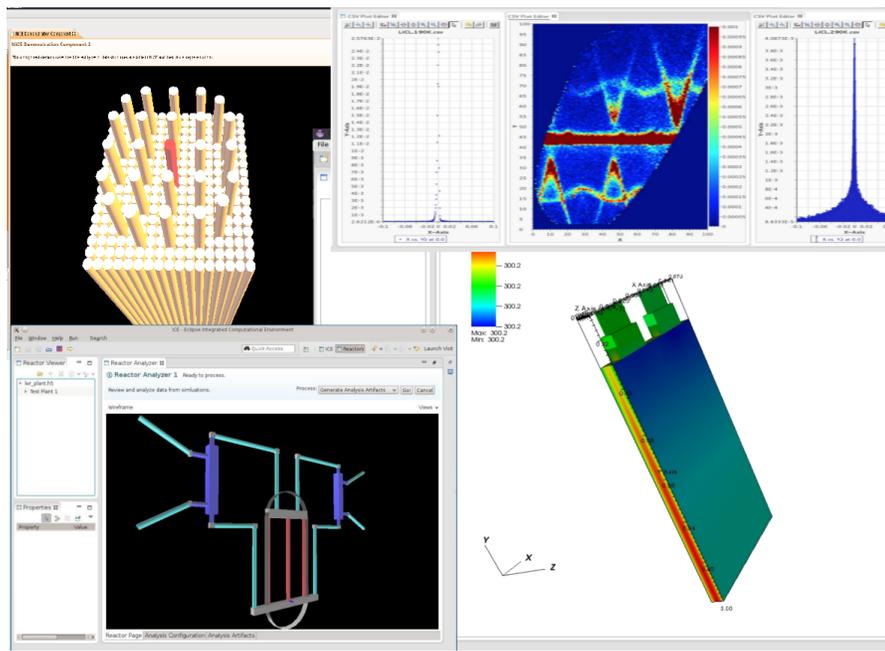
Science Highlight

Jay Jay Billings, Alexander J. McCaskey, Andrew Bennett, Jordan H. Deyton, Hari Krishnan, Taylor Patterson, Anna Wojtowicz

Introduction

Scientific Computing can be roughly divided into several different areas, one of which is that of Modeling and Simulation (M&S). Modeling and Simulation is a field that has existed, from a computing perspective, since World War II and finds its roots in government laboratories and academia. It is a field that encompasses both the art and the science of encoding the physical world into computers.

The subjects that have been investigated with computers are innumerable, with today's largest supercomputers simulating everything from cutting edge climate science to the behavior of matter at nanoscale resolution. However, while the subjects have changed and grown over the decades, many of the tools and the behaviors of the M&S community have not. A very large number of the most advanced computational studies of the physical world are done with tools that would be recognizable by a programmer from the late 1970s. The sophistication and the capability of today's software are much greater, but the tools - compilers, editors, third-party libraries, etc. - are not that different. That is to say that, on the whole, computational scientists are a command-line editing, low-level code loving kind of crowd!



(/community

/eclipse_newsletter/2015/january/images/visualization_tools.png)

Fig. 1 A collage of different visualizations tools in ICE. Clockwise from upper left: a nuclear fuel assembly, phonon data from nuclear scattering experiments, a model of nuclear plant, and a model of battery.

The Eclipse Platform has been adopted in many different fields and, in some cases, revolutionized the state of the art. Eclipse is not used in modeling and simulation as much as it is in other areas, but it is widely known. Several years ago we asked the question, "What if the tools in Eclipse that are used for *authoring scientific software* could be repurposed for *using scientific software*?" The result is the Eclipse Integrated Computational Environment (ICE); a general purpose, easily extended M&S platform that significantly updates the way computational scientists interact with their software. It improves productivity and streamlines the workflow for computational scientists when they need to create input files, launch jobs on local or remote machines, and process output. Since it is based on the Eclipse Rich Client Platform, developers can update it to support new codes or add new tools just as easily as users can perform simulations of nuclear fuels or visualize neutron scattering data.

Design and Implementation

ICE is designed around the idea that each task performed by a computational scientist - creating an input file, launching a job, post-processing data, etc. - can be done equally as well by an intelligent delegate that has the required data and knows the instructions necessary to perform the task. This seems broad, and it certainly is, but it works very well for the common tasks associated with M&S. For example, writing an input file is more about knowing *what* information should go in the file than *how* it should be written. Likewise, launching a massively parallel job is more about knowing *what* should be launched than the details of *how* to execute it. Each "intelligent delegate" in ICE is an Eclipse Plugin that inherits from an *Item* base class and provides a normal, pluggable OSGi service. In the ICE parlance, plugin and Item are used interchangeably most of the time.

ICE also includes some tools that are just, simply, tools and not part of the normal task-based design. Each case is directly related to manipulating or generating data, not using it, so some commonality does exist. These include tools, for example, for 3D visualization and editing ICE's materials database. Even though these tools are, in a sense, standalone tools, they are still integrated with the platform

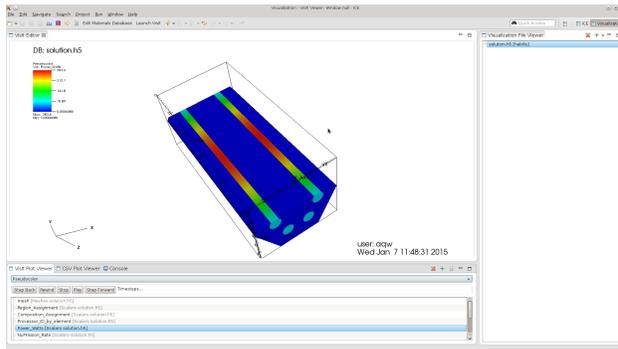
as OSGi services and used by the Items.

The entire platform is Eclipse-based with a small amount of C++ code for external I/O and real-time monitoring services for simulation codes. ICE does not use the C++ code itself; it is just provided to make it easier for the heavily C/C++ and Fortran based M&S community to communicate with ICE.

Examples: Nuclear Plant Simulations and 3D Visualization

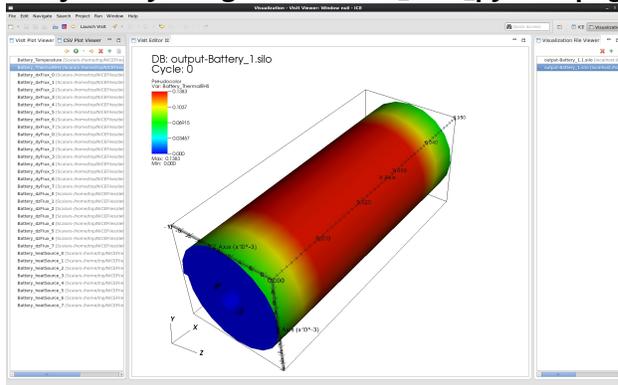
ICE's primary purpose is to make it easy to perform complicated M&S tasks. It has successfully integrated tools and simulation suites from across the U.S. Department of Energy (DOE) complex into a single, unified, cross-platform workbench for both "hard-core" computational scientists and those with limited experience. Hopefully, the following two examples, one each for visualization and input generation, will illustrate that.

The first example is visualization. 3D visualization is a critical, and arguably under utilized, tool in scientific computing. ICE provides tools that allow users to visualize their data using some of the most advanced visualization capabilities available, namely those of the VisIt suite. Users can perform custom analysis operations, such as slicing, via a built-in visualization console and other widgets. The view is completely interactive with rotation and zooming just a click away.



(/community/eclipse_newsletter

/2015/january/images/awesome_visit_python.png)



(/community/eclipse_newsletter

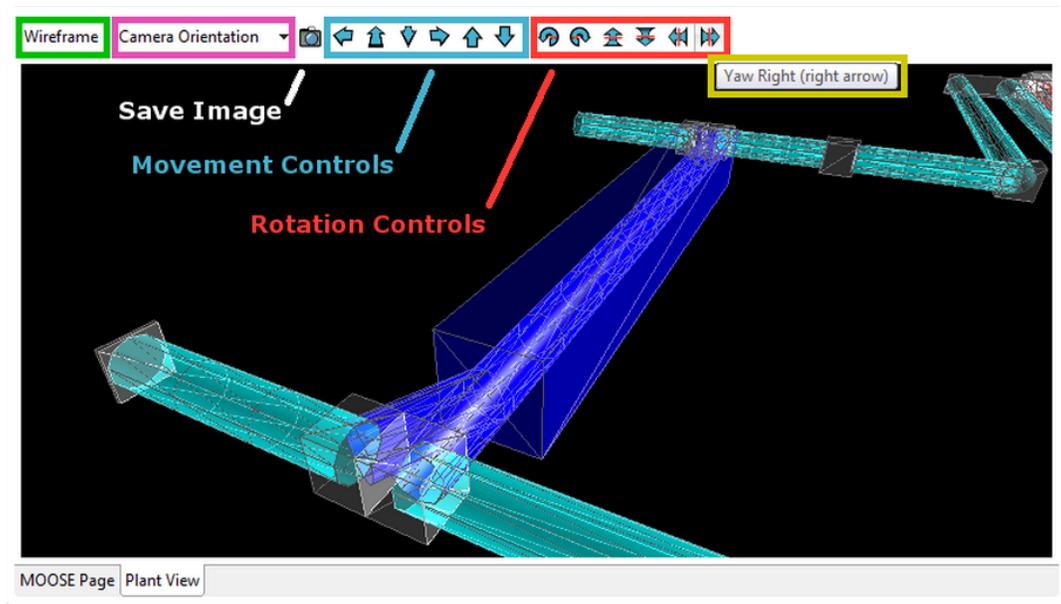
/2015/january/images/VisItICE.png)

Fig. 2 a) The power profile from a neutronics simulation in a sodium cooled fast reactor and b) the temperature profile of a simulated cylindrical cell battery, visualized in ICE.

Both of the images in Fig. 2. are from actual simulations and visualized in ICE. ICE's tools can connect to a local or remote running installation of VisIt, the latter making it possible to render very large visualization since VisIt has successfully rendered results from simulations with trillions of elements. Users can also connect to visualizations created by other users to share their their results or display

interesting properties in the output. Visit is one of a few different visualizations supported by the platform, all of which can be or will soon be accessible via OSGi services.

The second example is input generation. Input generation is one of the most difficult areas of M&S because dealing with the complex physics and complicated input formats are challenging at best. The challenge increases as the domain complexity increases, and one good example is the area of balance-of-plant or plant-level simulations for nuclear reactors. ICE has plugins for generating input for these simulations with the RELAP-7 simulator, based on MOOSE from Idaho National Laboratory, as well as launching RELAP-7 and viewing its results. Since the whole goal is to make everything very easy and push the state of the art, ICE takes this a step further by providing plant-level visualizations of the input.



(/community/eclipse_newsletter/2015/january/images/newfeatures.png)

Fig. 3 A plant-level view of a cooling loop in a nuclear reactor, as shown in ICE for RELAP-7.

In both cases, care has been taken to make it possible to reuse the new workbench extensions in other parts of the platform so that future releases will see embedded 3D visualizations in Eclipse Forms and plant-level views that are updated with simulation results in real-time, for example.

Availability and Contributions

Eclipse ICE is an Eclipse Technology project and the source code is available at <https://github.com/eclipse/ice> (<https://github.com/eclipse/ice>). The project is currently under active development by the Eclipse ICE team and review by the Eclipse IP team with the hope of joining a simultaneous release as soon as possible. Documentation on the project is available at <http://www.eclipse.org/ice> (<http://projects.eclipse.org/projects/technology.ice>) and <http://wiki.eclipse.org/ice> (<http://wiki.eclipse.org/ice>). Not all of the information about ICE is available on these pages since its initial contribution was only a few months back, but it is being migrated quickly.

The Eclipse ICE team welcomes contributions in any form from the community, (so long as they follow the Eclipse Contribution process, if it applies). The team hopes to double the number of core contributors and core contributing institutions in the next calendar year, so anyone with a good idea should speak up in the forums or on the Eclipse ICE users list, ice-users@eclipse.org ([mailto:ice-](mailto:ice-users@eclipse.org)

users@eclipse.org). Ultimately only a large, strong community of collaborating experts can bring a revolution to the way we use our scientific software!

Acknowledgements

We gratefully acknowledge the support of our sponsors in the U.S. Department of Energy, Office of Nuclear Energy, Advanced Modeling and Simulation (NEAMS); the Department of Energy, Office of Energy Efficiency and Renewable Energy, Computer-Aided Engineering for Batteries (CAEBAT) project; the Consortium for Advanced Simulation of Light Water Reactors (CASL); and the Oak Ridge National Laboratory Director's Research and Development Fund.

ABOUT THE AUTHORS



Jay Jay Billings

Oak Ridge National Laboratory (<http://www.ornl.gov/>)

- **Twitter (<https://twitter.com/jayjaybillings>)**

Copyright © 2015 The Eclipse Foundation. All Rights Reserved.