

## A Software Quality Engineering Case Study: The Jayenne IMC Project Gets a New Mesh Type

*Todd Urbatsch, Mike Buksas, Tom Evans, Aimee Hungerford, Scott Mosher,  
Chris Fryer, Jeff Densmore, Tim Kelley, CCS-2*

**S**oftware Quality Engineering (SQE) is a support activity for scientific methods and software development. End-user scientists sometimes understandably marginalize SQE, especially when SQE becomes a field of science or philosophy unto itself. However, the importance of software quality cannot be dismissed, especially when there are examples of experts' intuition being irreparably misshapen by bug-ridden software, or when, after decades and 10s of millions of dollars worth of development, a large software program is found to have a bug.

The Jayenne Implicit Monte Carlo (IMC) Project [1] is a collection of software in the Computer, Computational, and Statistical Sciences (CCS) Division that solves the thermal x-ray transport equations using the Fleck and Cummings IMC algorithm [2]. An application of the Jayenne Project software is shown in Fig. 1, where it is coupled to hydrodynamics to model the impact of a hot comet into a granite planet. It is the design of this software that makes testing and many SQE practices possible. The single most important design element is leveled design in which higher-level objects are built upon lower-level objects and there are no cyclic or same-level dependencies. Thus objects can be unit-tested and built upon with some confidence. At the top level of the Jayenne Project software, Milagro is the radiation-only code, and Wedgehog is the interface that can be utilized by application codes. Another design element is representing the independent variables of the mathematical equations as template parameters. For example, a Mesh Type (MT) template parameter represents the spatial variable, and any number of MTs can be built as long as they satisfy the interface requirements. One MT that the Jayenne Project lacked was a 3-D continuous

adaptive mesh refinement (AMR) MT. Here, we describe the testing that went with this new MT.

One tenet of ours is that testing must be repeatable and invocable automatically and on demand. Thus, unit tests are written and stored alongside the actual software. Unit tests verify that the software does what it is required to do. Within each object, or unit, we make use of Design-by-Contract (DBC) assertions that test data coming in, being used, and going out. These DBC statements, which will stop the code if testing fails, can be turned on or off at compile time for either debugging or performance, and they serve to document the requirements of the software. Higher-level unit tests verify ensembles of objects and interfaces. Integral tests, or highest-level unit tests, allow verification against analytic mathematical solutions. "Shunt" tests help verify package interfaces before they are integrated into application codes. Regression tests are simply all these tests monitored over time. The addition of the new 3-D AMR MT in the Jayenne Project software had over 300 unit tests and 30 integral tests at the Milagro level. The impact of the new MT on Wedgehog is shown in the unified modeling language (UML) diagram in Fig. 2, where affected components are shown in red.

A "Buggy Pageant" is a stunt we perform where we have someone plant bugs in our software and then we find the bugs in front of a live audience. One Buggy Pageant in 2006 pitted us against the most malicious bugs that Russian scientists from our sister lab, VNIIEF, could muster. The average time to find a planted bug is 13.3 minutes. We have since realized that Buggy Pageants would be just as useful as routine team activities whenever we add new capabilities. We performed two Buggy Pageants for the new 3-D AMR MT and

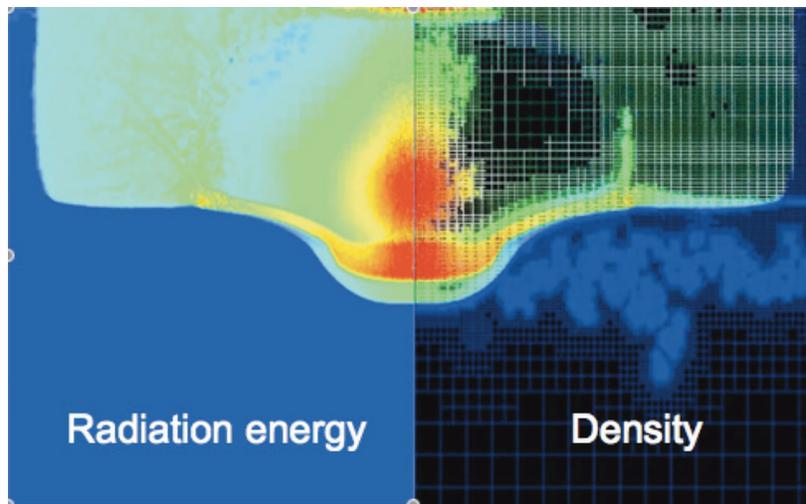
helped improve the testing of the MT and, in the process, educated the entire team in the new software.

*For more information contact Todd Urbatsch at [tmonster@lanl.gov](mailto:tmonster@lanl.gov).*

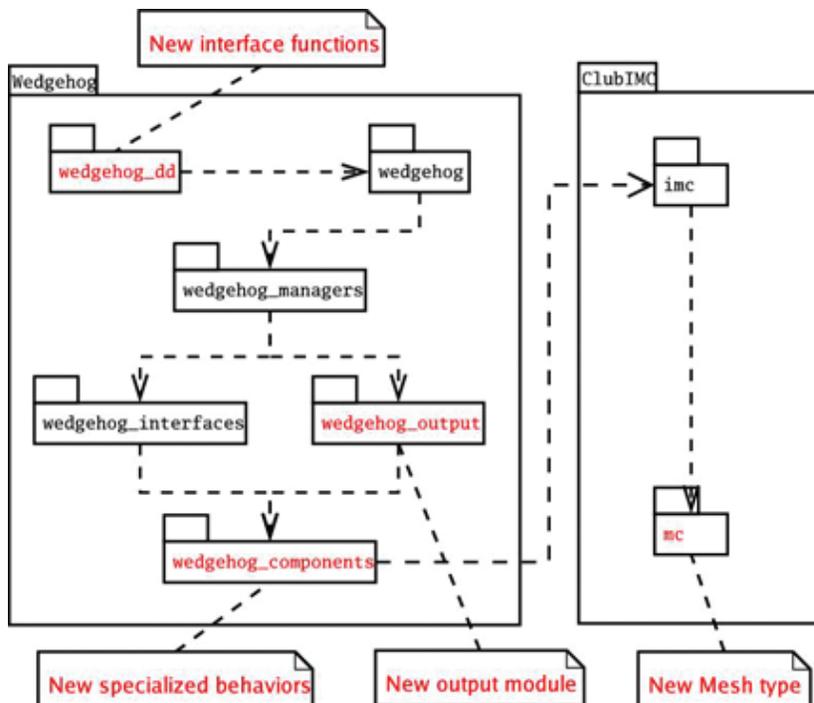
[1] T.J. Urbatsch, T.M. Evans, "Milagro Version 2: An Implicit Monte Carlo Code for Thermal Radiative Transfer: Capabilities and Usage," Los Alamos National Laboratory report LA-14195-MS (February 2006).  
 [2] J.A. Fleck and J.D. Cummings, *J. Comp. Phys.* **8**, 313–342 (1971).

**Funding Acknowledgements**

NNSA's Advanced Simulation and Computing (ASC), Integrated Codes Program Element, Transport Project.



**Fig. 1.** An application of the Jayenne Project software where it is coupled to hydrodynamics to model the impact of a hot comet into a granite planet.



**Fig. 2.** The impact of the new MT on Wedgehog in the UML diagram, where affected components are shown in red.