3.6 System Integration

Team Coordinators: Mark Brandyberry and Robert Fiedler

Research Engineer: Courtlan McLay

Research Programmer: Johnny Norris

Undergraduate Student: Jessie Hakes

Overview

Simulating solid propellant rocket motors requires solving an extremely complex, fully coupled multidisciplinary physics problem that spans a wide range of length and time scales. We have taken a staged approach in developing an integrated whole-system rocket simulation package, beginning with relatively simple physical models, geometries, and coupling in our first-generation code (GEN1), and progressing toward more detailed physics, geometries, and interactions in our second-generation code (GEN2). This year's efforts were devoted to integrating newly developed physics models into version 2.5 of the *Rocstar* rocket simulation package, verifying and validating the coupled code, performing several large-scale simulations, and designing and implementing *Rocstar* 3, which adds powerful new capabilities including advanced mesh modification schemes.

We improved the single-processor performance of *Rocface* by a factor of 15 on frost (the unclassified version of ASCI White) by avoiding the use of a few C++ language features. This makes frequent exchange of data between physics models much less expensive, and facilitates the performance of higher-accuracy simulations. The automatic profiling facilities in *Rocstar* 2.5 (in *Roccom*, part of the CS infrastructure) enable us to gather hardware counter performance data on various platforms. Comparing this data for compute-intensive portions of *Rocfrac* and *Rocflo* to a theoretical performance model, portions of these solvers were hand tuned, in-lined, or optimized using aggressive compiler optimizations, leading to an overall speedup of 22% for *Rocstar* 2.5 on a single processor.

RSRM Simulations

We compared results from a *Rocstar* 2.5 simulation of the Space Shuttle Reusable Solid Rocket Motor (RSRM) at 650 ms after ignition with scaled pressure data from an actual test firing known as QM-7 (Figure 3.6.1). The simulation agrees fairly well with the data, capturing the pressure drops at geometrical features such as joint slots. Agreement is not as good at the head end. One reason for this is that the data were obtained at a later time than was reached in the simulation, so regression of the propellant would have decreased the burning surface area there somewhat. The nozzle geometry in this simulation was considerably simpler than the real submerged nozzle, leading to larger discrepancies in that region as well.

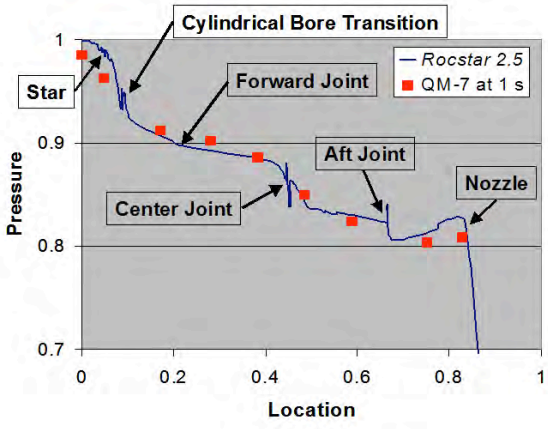

Fig. 3.6.1: RSRM axial pressure.

We recently completed a structured fluid mesh for the RSRM geometry used in the simulation described above, and partitioned it to run on 800 processors with the goal of speeding up RSRM simulations by a factor of four. At early times, a strong shock forms near the igniter outlet, which is difficult for the numerical scheme to han-

dle. Some experimentation with the time-dependent igniter mass injection boundary condition is required to overcome this problem.

We also created a new unstructured fluid mesh with a mix of tetrahedral, hexahedral, and pyramidal cells to improve resolution near domain boundaries. This mesh also includes a submerged nozzle and more detail of the inhibited surfaces in joint slots (see the cut-away image in Figure 3.62). This run requires an additional eight days of wall clock time on a machine like ALC to reach a quasi-steady state (~ 500 ms).

Fig. 3.6.2: Gas temperature at 42 ms in RSRM with submerged nozzle.

BATES Motor Simulations

The Ballistic Test and Evaluation System (BATES) motor is a laboratory scale US Air Force-sponsored rocket design that has been used for many years to experimentally investigate factors affecting motor performance. Data made available to CSAR include the precise geometry as well as the composition of various test propellants. Published data have also been obtained describing the results of a series of experiments to determine the effect of the propellant's aluminum composition on motor efficiency. This information will allow CSAR to validate the new particle transport and aluminum burning models in *Rocstar*.

The basic geometry of the BATES motor is shown in Figure 3.6.3. Figures 3.6.4 and 3.6.5 show results from a BATES simulation in which burning aluminum droplets of a single radius (30 micrometers) and aluminum fraction (90%; the rest is aluminum oxide) are injected beginning at time $t = 0$. The gas is initially in a quasi-steady state with no aluminum present. At 15 ms, a new quasi-steady state has not yet been reached. Gas and droplets injected in the narrow slot at the head end move rapidly along the axis of the rocket and out through the nozzle, but gas and droplets injected elsewhere on the propellant surface move at much slower speeds. As a result, the droplets near the axis of the rocket have shorter residence times, and therefore a larger final aluminum fraction, than do droplets that spend more time at larger radii. The smoke concentration is correspondingly higher at larger radii. Droplets entering the nozzle with a substantial amount of unburned aluminum reduce the efficiency of the motor.

Once the results with particles injected at a single diameter are fully verified and understood, we will perform simulations that include a model developed at CSAR of the agglomeration of aluminum droplets on the propellant surface before they are injected into the fluid domain. This represents a unique simulation capabil-

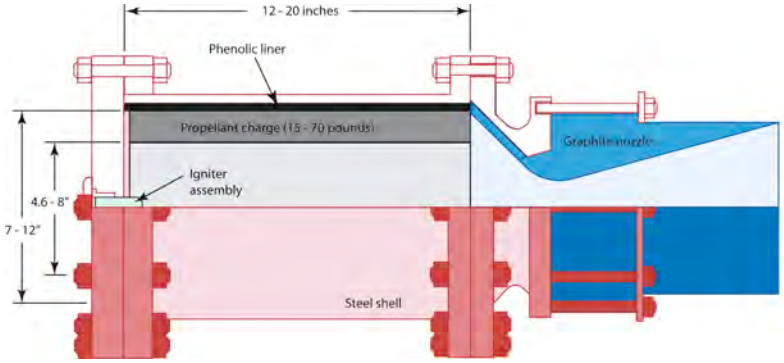

Fig. 3.6.3: BATES motor design.

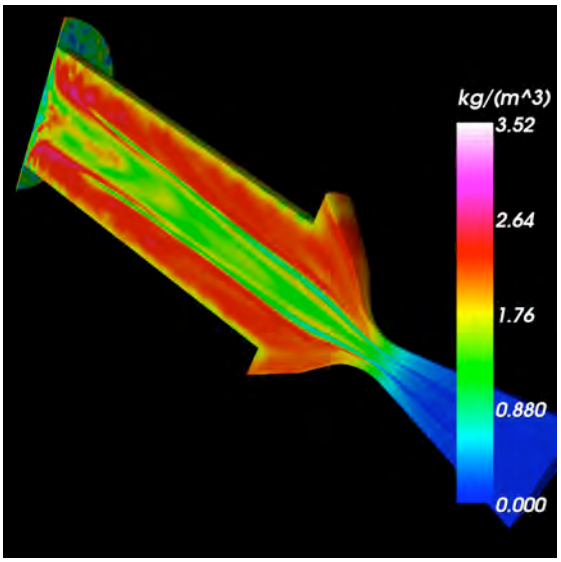

Fig. 3.6.4: Al_2O_3 smoke mass density in BATES motor at 15 ms.

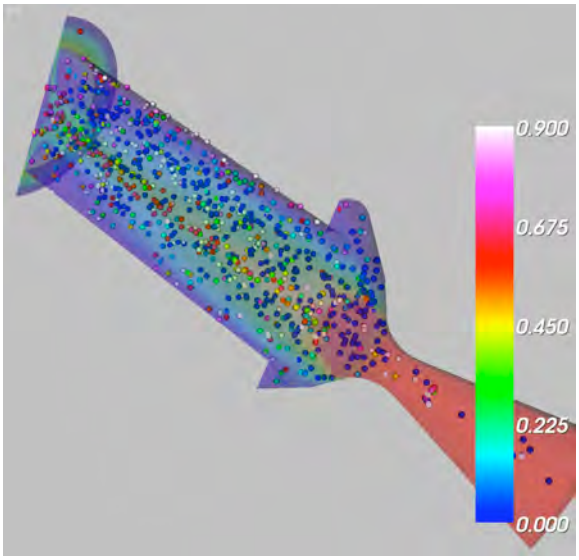

Fig. 3.6.5: Aluminum fraction in BATES motor at 15 ms.

ity of great interest to the rocket industry.

Titan IV

We used *Rocstar* 2.5 to improve upon our previous 3-D simulations of propellant slumping in the Titan IV SRMU Prequalification Motor #1. We are investigating an aeroelastic effect that occurs at joint slots that have, for manufacturing reasons, a smaller propellant bore on the aft side. The gas pressure is considerably higher inside the slot than it is just downstream of the slot, where the gas speed is much higher. This uneven load distribution tends to pull the propellant toward the axis of the rocket on the aft side of joint slots, possibly to the point of choking the flow of exhaust gasses. This effect caused the destruction of this motor on the test stand in 1991.

A new viscoelastic propellant material model, developed by Sofronis, includes the effect of deformation on the local porosity and was recently implemented in our implicit structural dynamics solver *Rocsolid*. In previous simulation we used our explicit solver *Rocfrac*. Figure 3.5.6 shows the porosity in a portion of the full domain near the joint slot 210 ms after ignition. The initial porosity value was 2% everywhere. At 210 ms, the propellant is under a compressive load and the porosity has decreased significantly in most locations. However, in the aft propellant section at the bottom of the stress-relief groove and where the propellant is joined to the case, the load will eventually become tensile and the porosity will increase, leading to further softening of the material and eventually catastrophic failure. This simulation terminated when the newly implemented global mesh motion scheme in the fluids solver *Rocflo* could no longer follow the deformation of the propellant. This mesh motion scheme uses Laplacian smoothing to propagate mesh motion

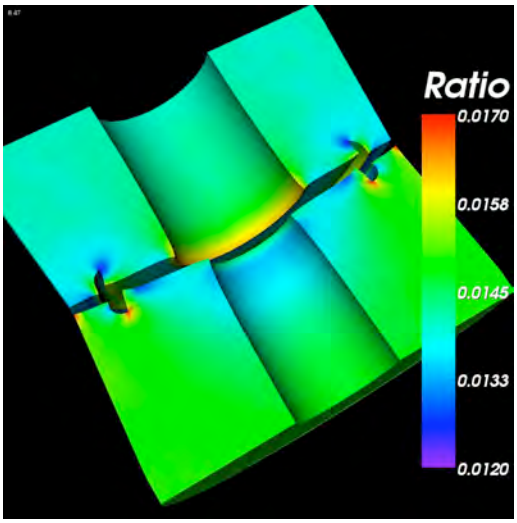

Fig. 3.5.6: Propellant porosity in Titan IV near joint slot at 210 ms.

from one mesh partition to its neighbors. Within each block, trans-finite interpolation is used to maintain a high-quality structured mesh.

Figure 3.5.7 shows the head-end pressure history from the Titan test firing, plus results from a series of simulations with increasingly sophisticated propellant material models. The simplest model is a rigid propellant, which results in a predicted head-end pressure that reaches a steady value near the expected operating pressure, since no slumping can occur. In a simulation with a linear elastic propellant model, which is most often adopted in the open literature, the area of the bore aft of the joint slot undergoes oscillations of increasing amplitude due to the aeroelastic effect described above. Such simulations lead to the conclusion that the pressure will eventually exceed the burst strength of the rocket case. Our next simulation uses the non-linear Arruda-Boyce constitutive model, which is appropriate for filled rubbery materials. This model captures the stiffening of the material, due to the presence of firmer particles embedded in the binder, as the strain exceeds more than a few percent. This simulation predicts pressure oscillations whose amplitude remains small, and therefore that the rocket does not explode, although the quasi-steady pressure is higher than the value expected in the absence of slumping.

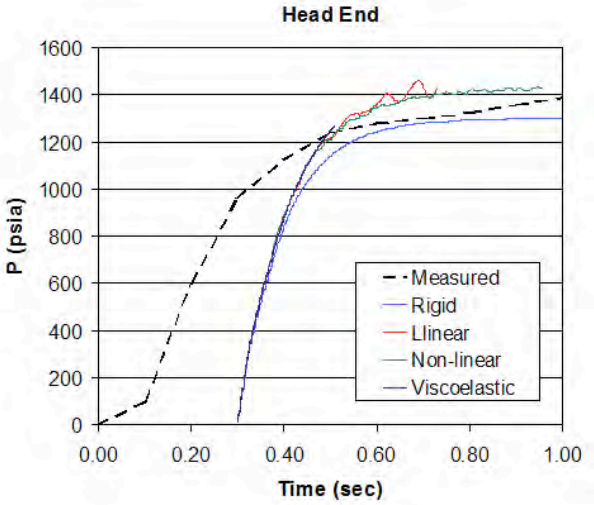

Fig. 3.5.7: Head-end pressure in Titan IV SRMU for various propellant models.

A viscoelastic material is characterized by a stiffness that is markedly higher under high strain rates than it is under low strain rates. In our most recent Titan simulation using Sofronis' model, the stiffness under a high strain rate is comparable to that of the simpler material models, but eventually decreases (with a relaxation time of 305 ms) by a factor of 10, enabling a much larger degree of slumping than the earlier simulations. At 210 ms, the slumping is too severe for the global fluid mesh motion scheme to follow, and the simulation terminates. We will repeat this simulation using the Mesquite package to smooth the fluid mesh in *Rocstar* 3.

Flexible Inhibitors

We performed fully coupled *Rocstar* 2.5 simulations ofgas flow in the vicinity of a flexible inhibitor protruding into the combustion chamber of the RSRM, including LES turbulence (dynamic Smagorinsky model). Figure 3.5.8 shows isosurfaces of the magnitude of the vorticity after a turbulent flow has been established. As the inhibitor swings back and forth due to gas pressure forces, the vortex shedding frequency falls and rises, introducing perturbations into the flow

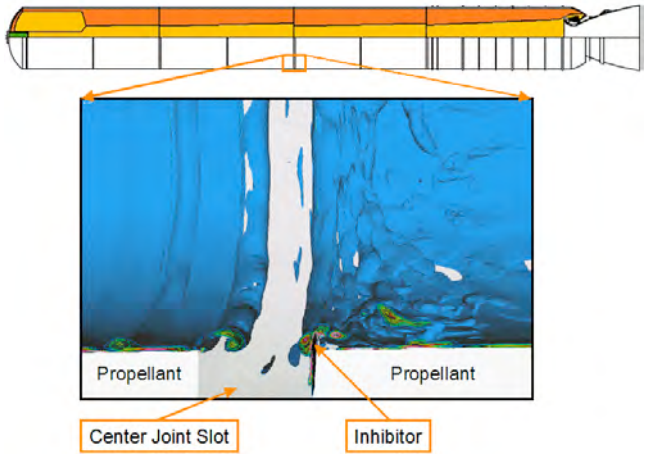

Fig. 3.5.8: Vortex shedding from flexible inhibitor at RSRM joint slot.

that could induce pressure oscillations and even drive instabilities. Another practical concern is enhanced burning aft of the inhibitor due to increased turbulent flow near the propellant surface.

In addition to the geometry shown in Figure 3.5.8, we also simulated RSRM configurations at 40 and 100 seconds after ignition. The latter time is of most interest to the rocket community, since the inhibitor protrudes by the largest amount. For this configuration, the inhibitor is deflected by a large angle, and the simulation terminates because the fluid mesh motion scheme is unable to handle that much deformation. We will repeat this calculation using Mesquite to smooth the fluid mesh in *Rocstar* 3.

Mesh Resolution Studies

We determined the order of spatial accuracy of our coupled scheme by comparing results from simulations with the same system time step but different meshes in the fluid and solid domains. Figure 3.5.9 shows the time-history of the average displacement magnitude of nodes on the burning propellant surface for simulations with 100k, 200k, and 300k total elements. Figure 3.5.10 shows the relative difference in average displacements for simulations on coarser meshes compared to the solution on the finest mesh. These three simulations can be used to determine the overall order of accuracy of the coupled scheme, which for this series turns out to be approximately 1 because the ALE algorithm in *Rocfrac* is only 1st order.

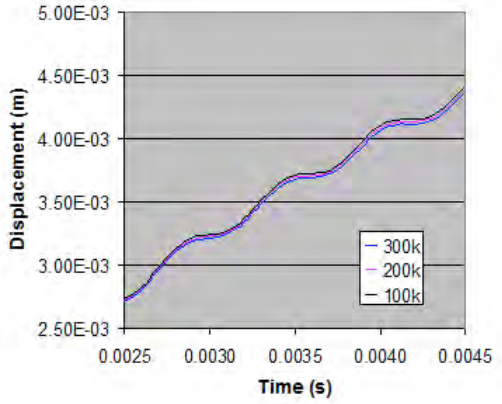

Fig. 3.5.9: Average displacement of nodes on propellant surface.

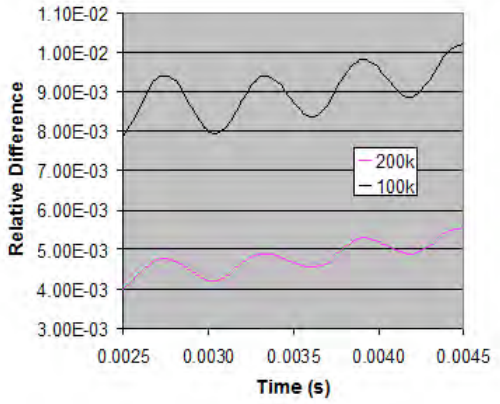

Fig. 3.5.10: Relative (to finest mesh) differences for solutions on coarser meshes.

Time Stepping Schemes

Over the past several months we have been studying temporal coupling schemes for multiphysics solvers based on the partitioned approach, that is, solvers comprised of several separate discipline-specific applications, as opposed to a single monolithic solver that simultaneously advances the entire set of evolution equations in each time step. Our goal is to ensure that our scheme is as stable and efficient as possible. We have identified a simple problem with a known analytical solution consisting of a 1-D fluid domain and a piston, which exhibits the type of instability that concerns us.

We have performed an exhaustive literature search of time stepping schemes and are investigating several strategies for improving stability and accuracy, including better predictors and predictor-corrector iterations. We have demonstrated that predictor-corrector iterations markedly improve the stability and accuracy of the explicit coupled scheme. Considerable analytical, numerical 1-D, and numerical 3-D analysis and experimentation are required to complete this investigation for both non-regressing and regressing interfaces.

Rocstar 3 Development

Development of *Rocstar* 3 is nearly complete. A diagram of the architecture is shown in Figure 3.5.11. *Rocstar* 3 is designed to handle much larger geometrical changes due to propellant burning and deformation than could *Rocstar* 2.5. Mesh modification services are provided by several commercial and institutional research packages, including Mesquite (Sandia National Labs) for mesh smoothing, YAMS/Tetmesh (Simulog Corp.) for complete surface/volume remeshing, and the local mesh modification tools from RPI/Simmetrix (Mark Shephard). These meshing tools are not parallel applications, so considerable effort is required to utilize them efficiently in large-scale simulations.

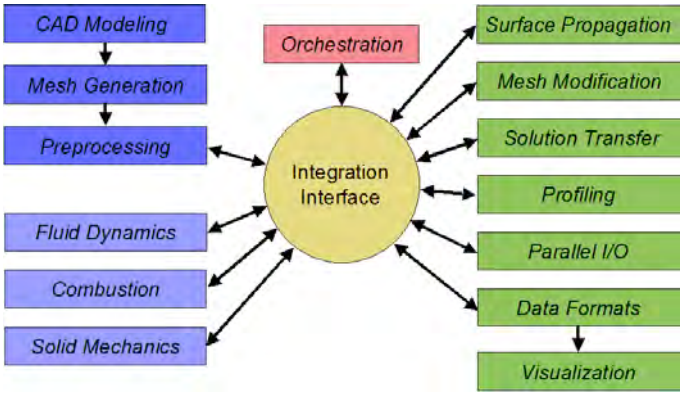

Fig. 3.5.11: *Rocstar* 3 architecture.

We have developed a method to smooth large meshes in parallel by allowing each partition to call Mesquite concurrently, and then exchange ghost cell information. We ensure that neighboring partitions agree on the locations of surface and ghost nodes, and that the resulting mesh quality is acceptable. Since the smoothing process does not change the number of nodes or their connectivity, no interpolation of the numerical solution onto the smoothed mesh is required because the ALE scheme in our solvers already takes arbitrary mesh motion into account.

For complete surface/volume remeshing, the entire mesh is first assembled onto a single MPI process (assuming the memory available to a single processor is sufficient). Without requiring user intervention, this single process invokes YAMS to create a new surface mesh, and then invokes TetMesh to create a new volume mesh. Next, the new mesh is automatically repartitioned for parallel execution, the numerical solution is transferred to the new mesh, and the simulation resumes. To facilitate restarting the code after remeshing, our new *Rocin* and *Rocout* modules enable our simulations (and input data prep tools) to read and write data files in any supported format, including HDF and CGNS, without requiring any changes to the physics component codes. These modules make it much easier for the physics modules to exchange data with other applications as well, including visualization tools such as *Rocketeer* and Tecplot.

Software Engineering Initiatives

Efforts to improve software engineering practices have continued over the past year. Communication, tracking, and control have all been enhanced either through improved group dynamics or through development and implementation of tools to facilitate best practices. Internal communications have been enhanced through a web site that provides access to the status of nightly builds, CVS code check-in logs, and code documentation. A new developer group, QUEST, has been formed to address software engineering, testing, and uncertainly quantification issues. Three members of the CSAR staff participated in the DOE ASCI Verification and Validation (V&V) meeting in San Diego during the past year, presenting results and initiatives from CSAR's V&V and software engineering programs.

Build and test practices have been expanded to include both the integrated code and several of the component stand-alone modules. Currently, both the integrated *Rocstar* code and the fluid modules are built on multiple platforms once a day with the *Rocbuild* system. Builds of the integrated code including the Charm++ package and various physics options are performed. Both the *Rocflo* and

Rocflu codes are built in a total of 36 different combinations on each of several machines to ensure timely identification of cross-platform compilation errors. Several machines that have recently become available to CSAR users are similar to some DOE ASC platforms. These machines are now part of the nightly build process to ensure that code changes will be compatible with the ASC platforms.

A small series of regression tests is also run using the *Roctest* module on one platform after each build to assess the stability of the most recent build. The regression tests provide confidence that the most recent code has not been seriously compromised by check-ins within the past 24 hours. An existing solution-differencing tool is being rewritten to use the *Rocin*/*Rocou*t I/O formatting services *Rocstar* 3. This code, called *Rocdiff*, will be integrated with *Roctest* to further automate the regression-testing package.

A substantial effort was made this year to check, clean up, and archive several of the CSAR problems sets, such as the scalability test, BATES motor, a small attitude control motor, and a lab scale motor. An archive of clean, QA'd datasets is being constructed for *Rocstar* 3. This archive will comprise the data source for our new *Rocprep* input data set preparation tool, which will automate the process of running a variety of preprocessing codes and then creating the directory structure for the dataset. However, initial grid construction, engineering analysis, and parameter specification cannot be automated.

Visualization

Support in the *Rocketeer* suite for ghost nodes and elements has been enhanced for unstructured meshes, and support for multiple connectivity tables for mixed meshes has been added. *Rocketeer* now reads data files in CGNS format (in addition to HDF) written by *Rocstar* 3, mesh generators, or other packages. CGNS promises more efficient access to particular data sets in a data file through its tree-like structure of nodes, which organize the data in a standardized fashion. An immediate benefit for *Rocketeer* users is the ability to visualize mesh quality measures in CGNS files written directly by the commercial meshing tool *Gridgen* before completing a *Rocstar* data set and running a simulation.

We have written two translators that utilize *Rocin* and *Rocout* to convert data files from our simulations into additional formats readable by third party meshing and visualization tools. These formatting services can also be used to produce an input mesh for *Rocfrac* from Gridgen output, which used to require its mesh data in a format written only by PATRAN. Thus, all meshes can be visualized by *Rocketeer* prior to running *Rocstar* 3.

Future Plans

Over the next two years, we will continue to upgrade *Rocstar* 3 as required to complete the 3-D coupled simulations described in detail in our Simulation Plan, including the BATES motor simulations described above. The mesh modification capabilities of *Rocstar* 3 will enable improved Titan and Inhibitor simulations. Milestones to be achieved in the coming year also include computing the full burn of a small rocket motor, which will fully exploit our advanced surface propagation and mesh improvement methods.

Regarding software engineering, we will integrate the *Rocbuild*/*Roctest* package with the *Rocdiff* comparison tool, and will increase the number of regression test problems to be run nightly. Additional test problems will be identified for verification and validation of *Rocstar*. All new and existing input data sets will be carefully updated, archived, and documented. The set of build and test platforms will be expanded to cover more target machines and OS types. A mobile *Rocbuild*/*Roctest* test kit will be developed for the DOE ASC platforms. While the kit will not be able to run in automated mode, it will allow on-demand building and testing of *Rocstar* on DOE machines.

Rocketeer will be extended to allow users to visualize the boundary conditions specified in files output by Gridgen and other commercial meshing tools, helping the user catch mistakes early in the *Rocstar* 3 input data set creation process.