Software for the Mu2e Experiment

Robert K. Kutschke
Mailstop #234, Fermilab, P.O. Box 510, Batavia, IL, 60510, USA
E-mail: `kutschke@fnal.gov`

Abstract. The Mu2e experiment at Fermilab is in the midst of its R&D and approval processes. To aid and inform this process, a small team has developed an end-to-end Geant4-based simulation package and has developed reconstruction code that is already at the stage of an advanced prototype. Having these tools available at an early stage allows design options and tradeoffs to be studied using high level physics quantities. A key to the success of this effort has been, as much as possible, to acquire software and customize it, rather than to build it in-house.

1. The Mu2e Experiment

Within the Standard Model, muons decay almost 100% of the time to final states that conserve lepton family number (LFN); the very rare branching fractions to final states that violate LFN are much too rare to be observed. Therefore any observation of an LFN violating muon decay is direct evidence for physics beyond the Standard Model. One such decay mode may occur when a negative muon is bound to an atomic nucleus, forming a muonic atom: coherent, neutrino-less muon to electron conversion in the Coulomb field of a nucleus. The final state is a mono-energetic electron plus an unobserved, recoiling, intact nucleus; all background processes produce electrons with a continuous energy spectrum. The Mu2e experiment will form muonic Aluminum, measure the energy spectrum of electrons from its decay and ask if an excess is observed at the conversion energy, about 105 MeV. The measured quantity in the Mu2e experiment is

$$R_{\mu e} \;=\; \frac{\mu + Al \rightarrow e + Al}{\mu + Al \rightarrow \text{all nuclear captures}}. \qquad (1)$$

The goal of the experiment is to achieve a single event sensitivity corresponding to $R_{\mu e} = 5.4 \times 10^{-17}$, which is about 4 orders of magnitude better than the previous best upper limit of $R_{\mu e} < 7 \times 10^{-13}$ at the 90% CL [1].

Figure 1 shows the major elements of the Mu2e apparatus, including the muon beamline and the detector elements. An 8 GeV proton beam enters from the right and strikes the production target, producing pions that decay to muons. The muon beamline is formed by a system of three graded-field solenoids and their associated collimators; the muon beamline captures the backwards going muons, reduces the flux of unwanted particles, and directs the muons onto the muon stopping target, a set of thin Al foils. Most muons that reach the stopping target range out in these foils and are captured to form muonic atoms. Electrons from the decay of the muonic atoms are directed by a graded magnetic field onto a tracking system and an electromagnetic calorimeter. The tracking system makes a precise ($\sigma_p/p \simeq 0.1\%$) measurement of the momentum of charged particles that traverse the tracker. The calorimeter serves two purposes: together the

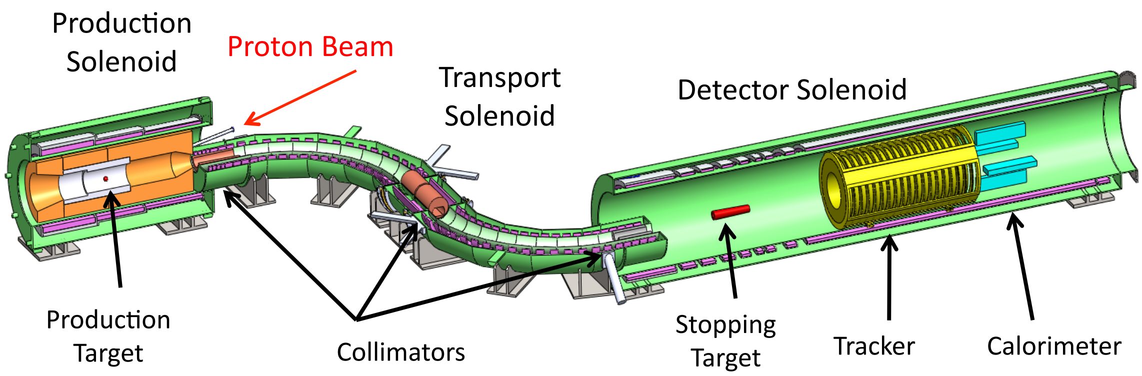

Figure 1. The major elements of the Mu2e muon beamline and detector; details are given in the text.

tracker and calorimeter can reject several rare, but critical, background processes that cannot be adequately rejected by the tracker alone; the calorimeter also provides a redundant trigger for testing the primary, trigger-less data acquisition mode. Not shown in the figure are a system to monitor the extinction of the proton beam[1], a Cosmic Ray Veto system (CRV) and a system to measure the spectrum of X-rays emitted by muons as they cascade to the atomic K-shell; this last measurement is the primary input into the measurement of the denominator in $R_{\mu e}$.

Additional details about the physics of the Mu2e experiment and the design of the Mu2e apparatus can be found in the Mu2e Conceptual Design Report (CDR)[2] and in other documents found on the collaboration web site[3].

2. Offline Software

The Mu2e offline software comprises codes to simulate the experiment, calibration codes, reconstruction codes, analysis codes and codes to characterize the quality of results delivered by the reconstruction codes when they are run on simulated events. At the present stage of development, the reconstruction codes work only on simulated events but they are being written so that they will work, as is, on experimental data once that is available.

Studies undertaken to date include characterizing the joint spatial, temporal and momentum-space population of the particle fluxes entering the DS, including the μ^- flux and the fluxes of all interesting background species; understanding how these fluxes depend on the designs of the magnetic field, the production target, and the collimators; computing realistic distributions of non-signal hits in the detector subsystems; computing the efficiency for reconstructing conversion electrons in the presence of the non-signal hits; computing the power of the detector to reject signal-like backgrounds produced by non-signal processes; computing the radiation and heat loads on elements of the apparatus; and redoing all of the above for variants on the design of the experiment.

For these studies, Mu2e has used four different software packages. G4beamline and FastSim permitted a fast start to many critical studies but they do not have all of the features needed for development of hit based reconstruction codes. The art based Offline software, Mu2eSim and Mu2eReco, does have the full feature set needed for hit based simulations and reconstructions; however these tools took longer to develop, during which time the insights gained from G4beamline and FastSim were critical. Finally, MARS is used when precision

[1] The proton beam arrives in bunches with a full width at the base of about 200 ns; the experiment requires that, for every proton within a bunch, fewer than 10^{-10} protons arrive between bunches. This ratio is known as the *extinction* of the proton beam.

information about low energy neutrons is required. Each package is described in its own sub-section, below.

2.1. G4beamline

Most studies of the Mu2e muon beamline have been performed using G4beamline, "a single-particle tracking program based on the Geant4 simulation toolkit. It is specifically designed for the simulation of beamlines." [4] The Mu2e experience is that G4beamline provides, as advertised, an easy-to-learn interface that enables effective use of Geant4 by those who are not trained to use it in its native form.

The essence of G4beamline is that it provides a user friendly layer on top of Geant4, allowing the user to think in terms of beamline elements without needing to know the intricacies of Geant4. The input to G4beamline is a file that describes the elements of the beamline, including the placement of materials, the source of primary particles, the magnetic field maps, and a set of virtual detectors. The input file also controls the behaviour of Geant4 by choosing a physics list, the values of user cut-off parameters and the algorithm used to integrate the equations of motion in a magnetic field. The operation of G4beamline is to digest the input file, instantiate the Geant4 geometry, instantiate Geant4 sensitive detectors for each virtual detector, shoot primary particles, one per event, and invoke Geant4 to track through the apparatus the primary particles, plus any descendants they may produce. At the end of each event, G4beamline summarizes in several ROOT ntuples, the position, time, direction and particle type of selected particles that pass through each virtual detector specified in the input file. G4beamline can also be configured to record every G4Step of selected particles.

Using G4beamline Mu2e physicists had a quick start studying the properties of the muon flux arriving at the stopping targets and the properties of the many backgrounds that accompany those muons. G4beamline has also been used for studies of the heat and radiation shield, an element of the Production Solenoid (PS) system that protects the cryogenic magnet coils from the particle spray produced when protons strike the production target.

There are several aspects of G4beamline that limit its applicability for detector studies in general and for more detailed studies of the muon beamline. The G4beamline geometry language does not provide access to the full set of solids defined by Geant4; for example boolean solids are not supported. In addition G4beamline does not keep a detailed parent-child history of what happened inside Geant4, which complicates the understanding of some features of the simulated events. Nor does G4beamline provide the infrastructure needed for the creation of realistic hits. All of these features are available inside Mu2eSim, described in sub section 2.3.

In summary, G4beamline has enabled a fast start and a continued fast pace in the design of the muon beamline; this work was done by physicists who understand the physics well but who have no training in either C++ or Geant4.

2.2. FastSim

The FastSim program[5] is a fast, surface-based, simulation and reconstruction tool originally developed by the *BABAR* collaboration for use when very large Monte Carlo samples were needed, for example, when validating high-dimensioned, unbinned maximum likelihood fits. Among many other features, this package includes the battle-tested *BABAR* Kalman filter code. The *BABAR* collaboration has given Mu2e permission to use FastSim. FastSim was next used as a design tool for the SuperB experiment and the Mu2e code is a ported from the SuperB port.

As used by Mu2e, FastSim processes each event in three steps. First it simulates the response of the detector to a signal track; next it overlays frames of background hits; finally it fits the simulated track using the Kalman filter. The end result is a set of track parameters and their covariance matrix, each valid in the neighborhood of suitably chosen reference surfaces along the trajectory.

The simulation step includes transport in a (possibly non-uniform) magnetic field, multiple scattering, energy loss, and bremsstrahlung in the detector materials; the models of scattering and energy loss include include non-gaussian tails. When a simulated track passes through a measurement surface, FastSim creates a hit, which is smeared by a function with a gaussian core plus non-gaussian tails. A model of hit inefficiency can be applied at this time. Hits for background frames are generated in a similar way. The result of the first two steps is an event containing hits from both the signal and background sources.

The reconstruction step begins by using the Monte Carlo truth to identify the hits that belong on the signal track. FastSim then looks along the true trajectory to identify hits from background sources that lie close to the true trajectory; it has a model, tuned to *BABAR* data, of how often an incorrect hit from this sample should be added to the track and how often it should replace a correct hit. The set of all hits believed to be on the track is then passed to the Kalman filter which is configured to reject outlier hits.

The result of this process is an ensemble of fitted signal tracks that have tails containing contributions from the tails in multiple scattering, tails in energy loss, tails from bremsstrahlung, tails in the straw hit spatial resolution function and tails due to an important class of pattern recognition errors. This ensemble of tracks was used to characterize the efficiency and momentum resolution of the Mu2e tracking system. This job was repeated for variations of the design parameters of the tracker: the number of stations, the spacing between stations, the thickness of the straws, the resolution each straw and so on. FastSim was also used to study the geometric acceptance of candidate calorimeter designs.

Finally, FastSim was used to study how errors in the modeling of the magnetic field might give rise to biases and tails in the momentum resolution function. For this study, FastSim integrated, along the trajectory, the shift in the momentum caused by the error in the field. This technique permits a robust estimator of effects that are too small to be seen directly by looking at the difference between generated and reconstructed quantities. This study was performed to compute the required precision with which the magnetic field must be mapped within the tracking volume.

Mu2e made two major improvements to FastSim. First, when using the Mu2e geometry, which is quite different from that of *BABAR* or SuperB, the FastSim navigation code did not correctly deal with tracks that turn through multiple arcs of their helix; this has been fixed. Second, the non-gaussian tails in the energy loss and scattering distributions were tuned to *BABAR* data; however some of the materials in the Mu2e detector are much thinner than those in the *BABAR* detector and the models were discovered to be outside of their range of validity. These models were improved to work correctly on thin materials.

Because FastSim uses a surface based (2D) description of detector materials, it does not have the infrastructure needed to create realistic hits and so cannot be used as input for true hit-based pattern recognition. FastSim is no longer heavily used but it played a most valuable role in the early design studies.

2.3. art Based Offline Software: Mu2e Sim and Mu2eReco

When the Mu2e effort at Fermilab began, the collaboration requested that the Fermilab Computing Division (CD) supply and support infrastructure software for use by the collaboration. The term *infrastructure software* includes a framework (the entity that loads the dynamic libraries and drives the event loop), an event-data model, a persistency mechanism, run-time configuration, maintenance of the state of random number engines, message logging and tools for the management of singleton-like entities such as geometry and conditions data. It also includes the tools to manage access to data bases, file catalogs and the GRID workflow management tools. Mu2e required one system that could be used for all purposes, from the lowest level non-real-time task in the trigger/DAQ system through to final analysis; the steps in between include reconstruction, calibration, filtering/skimming of data sets, and simulation. Moreover the system had to be an analysis-first design, not a simulation-first or a graphics-first design.

A team from CD recommended that Mu2e adopt a fork of the CMS infrastructure software (CMSSW), a choice that would leverage CD's considerable investment in the development of CMSSW. Some of the important ideas retained from CMSSW are the notions of modules, services, data products, the ROOT-based persistency mechanism, parameter sets for run-time configuration, the message logger and the patterns by which these entities interact with each other. Mu2e anticipates that its conditions data will be sufficiently simple that adequate functionality can be delivered using a service based model; therefore EventSetup was removed. Most other features of CMSSW were retained. Additional details are available[6][7].

Mu2e agreed to this plan and, in early 2009, started to use the first generation fork from CMS[2]. In this environment Mu2e learned how to use the combination of Geant4 and the framework, developed its first generation of data products to hold the outputs produced by Geant4, developed its first analysis modules and delivered the first, preliminary physics studies. The interface to Geant4 uses the recommended technique of inheriting from G4RunManager and overriding the BeamOn method; in this way the framework drives the event loop and Geant4 behaves like just another module that processes one event at a time.

A key design decision is that Mu2e code does the minimal possible Mu2e-specific work inside Geant4; it simply exports information produced by Geant4 to the framework event object. It exports a parent-child history of every particle tracked by Geant4 and, for every sensitive detector, it exports a summary of the G4Step information for every step that is taken in that sensitive detector; the G4Step summary class is named StepPointMC. Optionally the full trajectory information can be exported for every G4Track; this last feature permits the event display to draw complete simulated events.

Hit formation is done in framework modules, one for each detector type, that are scheduled to run after Geant4 has completed each event. Hit making codes read StepPointMC objects and combine one or more to form objects that represent data-like hits; the Monte Carlo truth information for the hits its stored in separate data products, which obeys the rule that objects that can exist in data events must not contain any Monte Carlo truth information.

The Mu2e geometry is rich in virtual detectors; in addition, many non-traditional elements, such as the stopping target foils, are instrumented as Geant4 sensitive detectors. Together these elements allow one to perform detailed studies at the Monte Carlo truth level and they provide the information needed to develop, debug and characterize the reconstruction algorithms. Mu2e is aware of the issues caused in CMS by the nuclear counter effect; to address this the Mu2e geometry explicitly represents the readout devices attached to each calorimeter crystal and instruments these as a separate class of sensitive detectors. The calorimeter hit making code will produce a saturated readout value if a readout device is hit directly by a particle.

By spring 2011, the CD team had evolved the first generation fork into the product now called art[6][7], which is used today by Mu2e, NOvA, ArgoNeut, MicroBoone and the muon g-2 experiment. Averaged over these two years, the level of effort going into art was less than 2 FTEs and the level of effort going into the Mu2e code was about 1 FTE. Even with this low level of effort, by spring 2011, a crude but complete hit based simulation and two event displays, one using the Geant4 display technology and a ROOT based display that draws geometric information plus many of the data products that can be found in an event,

Soon after the first release of art, the art team added several new features. For example, art supports persistable pointers that reside in an object in one data product and point to another object in an arbitrary data product. This technology is now in wide use within Mu2e data

[2] This fork was informally named CMS-lite, which is not to be confused with a very different product of the same name within CMSSW.

products; it is used to record Monte Carlo truth information and the calorimeter cluster code uses it to record which hits are members of each cluster. Art also supports data products that describe relationships between objects in other data products; this last feature will be used for things like recording which tracks from the tracker match to which clusters of energy in the calorimeter; the technology supports 1:1, 1:many, many:1 and many:many matches. Among other things, this technology will also be used for expressing matches between simulated and reconstructed particles.

Working closely with Mu2e, the art team developed event mixing code to overlay pileup events on top of signal events. In Mu2e, a typical signal event is accompanied by a few hundred events from a half dozen or so separate pileup streams. Most pileup events make only 1 or 2 hits in the detector so the occupancy is manageable but the spatial and temporal structure of the pileup must be well modeled; therefore the pileup events are generated by long runs of the simulation using background event generators. The pileup problem was carefully factorized into two pieces: the person who knows about the details of the persistency model need know nothing about the content of the data products and the person who knows about the contents of the data products need know nothing about the intricacies of persistency. The solution is implemented as a mixing module template, which knows about persistency, plus a user supplied detail class that provides any required knowledge of the contents of the data products. Each mixing module is an instantiation of the mixing template with the detail class as the template argument; methods in the detail class are called by the mixing template to perform dataproduct-specific specific tasks. This solution places no limit on the number of input mixing streams or on the number of pileup events read from each mixing stream — the only limits are imposed by the available memory on the computer on which it is running. The Mu2e mixing detail method class knows how to update the persistent pointers for the new data product geometry and how to shift events in time.

It was mentioned above that Mu2e exports StepPointMC objects from Geant4 and forms hits in a separate art module. When event mixing is done, the mixing is done at the StepPointMC level and hit formation uses the ensemble of the many StepPointMCCollections. In other words, event mixing is done at the analog level, not at the level of digitized hits.

Mu2e has ported the FastSim Kalman filter (originally the *BABAR* Kalman filter) to run in the art environment; this operation took only a few weeks and it exploits the many person years of effort that have gone into the development of that code. Over the course of the past year, Mu2e collaborators have developed tracker pattern recognition code that operates on events with pileup; once a track candidate has been found it is passed to the Kalman filter for final fitting, including outlier rejection and the possible addition of hits that are near to the trajectory but not included in the initial hit list. Current work has been focused on the improvement of the left-right ambiguity resolution in the straws. All of this work uses only information that will be available in data; the Monte Carlo truth is carefully segregated and is only used for quality control after the reconstruction is complete.

Figure 2 shows the results of a simulation for events that contained one conversion electron, plus the appropriate pileup. The pattern recognition and Kalman filter code was run on these events. For each reconstructed track, the measured momentum was determined at the upstream face of the tracking system. After the fit was completed, the Monte Carlo truth momentum at the upstream face of the tracker was extracted from one of the virtual detectors. Figure 2 shows the difference, measured-true momentum, evaluated at the upstream face of the tracker. The tails are asymmetric because electrons that bremsstrahlung in the tracker material are preferentially reconstructed with a low measured momentum. Figure 3 shows the same quantity for slightly tighter track quality cuts; this significantly reduces the high side tail at the expense of reducing the efficiency from 46% to 38%. These plots serve as a proof of principal that an adequate reconstruction algorithm exists, despite the unusual tracker geometry.

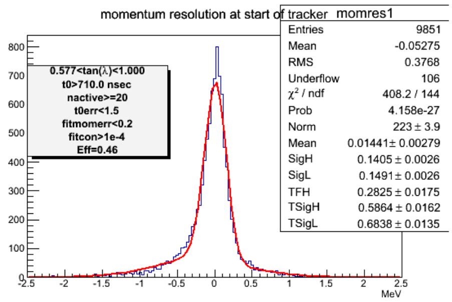

Figure 2. Momentum resolution of the tracker, with loose cuts.

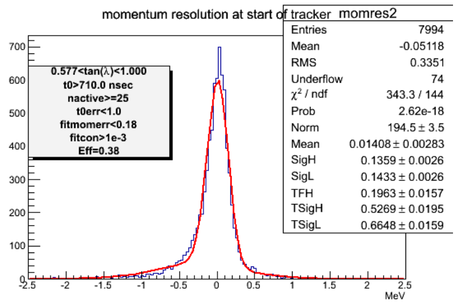

Figure 3. Momentum resolution of the tracker, with tighter cuts.

At this time efforts are underway to form clusters in the calorimeter, to match tracks to clusters and to extrapolate tracks through the inhomogeneous magnetic field from the tracker to the stopping targets.

In summary, the track reconstruction code is now at the level of an advanced prototype and the first generations of the other reconstruction codes are expected soon. The two major acquisitions, art and the FastSim Kalman filter, have allowed most of the Mu2e physicist effort to go towards physics software, not infrastructure software. In this sense the art based software has been a great success. In the future many studies will be moved from G4beamline to Mu2eSim in order to exploit the greater level of detail recorded by Mu2eSim.

2.4. MARS

The authors of MARS describe their code as "... a Monte Carlo code for inclusive and exclusive simulation of three-dimensional hadronic and electromagnetic cascades, muon, heavy-ion and low-energy neutron transport in accelerator, detector, spacecraft and shielding components in the energy range from a fraction of an electron volt up to 100 TeV." [9]. Mu2e uses MARS configured so that it uses the MCNP [10] package for low momentum neutrons. It is understood throughout the HEP community that MARS plus MCNP is the best calibrated code for simulation of neutrons, in particular neutrons with low kinetic energy.

Two MARS experts work on Mu2e and have used MARS to study radiation levels throughout the experimental hall. This informs the requirements for shielding, for radiation safety and for the radiation hardness of electronics. The team is currently computing the neutron flux through the cosmic ray veto counters; if this flux is safely low, then it will be possible to retain the baseline design of a plastic scintillator based CRV system; if the flux is too high the backup plan is to move to gas based detectors, which are less sensitive to neutrons. MARS was also used for the critical calculation of the heat and radiation load in the coils of the PS; this is used to study variants on the design of the heat and radiation shield. This calculation was sufficiently important that it was done with both MARS and G4beamline; this was done as a cross-check on the underlying physics models and cross-section tables.

The most critical calculations will also be done using Mu2eSim. Because these three codes were developed independently, the three way cross-check will help expose any pilot error that has escaped other controls.

Until recently, security requirement imposed by the MARS and MCNP license issuers restricted the running of these codes on insecure platforms. Working with the MARS team, Mu2e worked through the security issues and pioneered a procedure for running MARS and MCNP on GP Fermigrid. This procedure is now used by many Fermilab Intensity Frontier experiments.

As for G4beamline, MARS does not make output that can easily be used as input to a hit level simulation; that job is done by Mu2eSim.

3. Geometry Issues

One weakness of the present system is that each package has its own geometry language that is inconsistent with that of the other packages. To address this, there is a plan to develop a common authoritative source from which all of the geometry descriptions are derived.

4. Summary and Conclusions

A small team has made rapid progress in the development of software for the Mu2e experiment: an end to end simulation and advanced prototype track reconstruction codes are in place. Calorimeter and Cosmic Ray Veto reconstruction codes are in progress. Rapid progress was enabled by leveraging the work of others: as much as possible the infrastructure software has been acquired and configured, not built.

Acknowledgments

The author would like to thank the Fermilab management and staff for their strong support of the Mu2e experiment. This work is supported in part by the Office of Science of the U.S. Department of Energy, the U.S. National Science Foundation, INFN in Italy, and the U.S.-Japan Agreement in High Energy Physics.

References

[1] U. Bellgardt et al. (SINDRUM Collaboration), Nucl. Phys. B299, 1 (1988).
[2] http://mu2e-docdb.fnal.gov/cgi-bin/RetrieveFile?docid=1169;filename=CDR%20Final.pdf;version=14
[3] http://mu2e.fnal.gov
[4] http://g4beamline.muonsinc.com
[5] FastSim: A Fast simulation for the SuperB detector. R. Andreassen, M. Sokoloff (Cincinnati U.), N. Arnaud, L. Burmistrov (Orsay, LAL), D.N. Brown, J. Carlson, I. Gaponenko, A. Suzuki (LBL, Berkeley), C.-h. Cheng (Caltech), A. Di Simone (Rome U.,Tor Vergata & INFN, Rome2) et al.. Dec 2011. 6 pp. Published in J.Phys.Conf.Ser. 331 (2011) 032038
[6] https://cdcvs.fnal.gov/redmine/projects/art
[7] C. Green and J. Kowalkowski and M. Paterno and M. Fischler and L. Garren and Q. Lu, "The **art** framework", in these proceedings. talk for abstract 354 by Chris Green in Event Processing (track 2), Monday afternoon.
[8] "Taking Global Scale Data Handling to the Fermilab Intensity Frontier"; Adam L. Lyon, Robert A. Illingworth, Marc Mengel, Andrew J. Norman, in these proceedings.
[9] N.V. Mokhov, "The Mars Code System User's Guide", Fermilab-FN-628 (1995); N.V. Mokhov, S.I. Striganov, "MARS15 Overview", in Proc. of Hadronic Shower Simulation Workshop, Fermilab, September 2006, AIP Conf. Proc. 896, pp. 50-60 (2007); http://www-ap.fnal.gov/MARS/
[10] MCNP-A General Monte Carlo N-Particle Transport Code, Version 5, LA-UR-03-1987, LA-CP-03-0245, LA-CP-03-0284 (2005), http://mcnp-green.lanl.gov