



BNL-102621-2013-IR

ADVANCE System
Requirements

David A. Brown, Ramon Arcilla, Michal Herman

National Nuclear Data Center
Brookhaven National Laboratory
Upton, NY 11973

July 15, 2013

Energy Sciences & Technology Department
National Nuclear Data Center
Brookhaven National Laboratory
P.O. Box 5000
Upton, NY 11973-5000
www.bnl.gov

Notice: This manuscript has been authored by employees of Brookhaven Science Associates, LLC under Contract No. DE-AC02-98CH10886 with the U.S. Department of Energy. The publisher by accepting the manuscript for publication acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes.

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or any third party's use or the results of such use of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof or its contractors or subcontractors. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

ADVANCE System Requirements

Authors: David A. Brown, Ramon Arcilla, Michal Herman
National Nuclear Data Center
Brookhaven National Laboratory
Upton, NY 11973

Date: July 15, 2013

Revision: 3

0. Introduction

The NNDC requires a continuous integration/continuous deployment (CI/CD) system as part of its effort to automate and simplify our workload and to ensure software and data quality (especially for the ENDF/B series of nuclear data libraries [ENDF]). A continuous integration system is one that continually monitors a software repository (e.g. a subversion repository), extracts changes and builds and tests all changed components. A continuous deployment system takes the results of the integration step and publishes them to, for example, a website. CI/CD systems are a common tool for automated software Quality Assurance (QA) during software development. CI/CD systems support agile development as well as more traditional approaches to software/data development.

ADVANCE is a CI/CD tool for software and data QA and builds on off-the-shelf (e.g. RunDeck [RunDeck] or BuildBot [BuildBot]) functionality. For software projects, not much additional functionality is needed. For data projects, the changes to the core functionality are substantial.

Although ADVANCE has just undergone a release (v0.7) [ADVANCE], it is due for a major release once it includes support for code projects, distributed build and data/code benchmarking. This document will serve as a guide for this and future developments to ADVANCE. The final section of this document is a product review of three CI/CD systems which we are considering for use in the next major ADVANCE release.

In this document, we focus mainly on testing libraries written in the legacy ENDF format. As we move to new nuclear data format, we anticipate will need to revisit this document.

I. General System Requirements

Here we detail general system requirements for all types of software/data projects. Most off-the-shelf CI/CD systems meet these general requirements.

1.a. Project management requirements

1. **Version controlled.** The system must interact with a GForge [GForge] subversion [subversion] or git [git] project. It must either receive push notifications from subversion or poll subversion, looking for changes, pull down change set and run all defined tests. This is illustrated in Fig. 1.

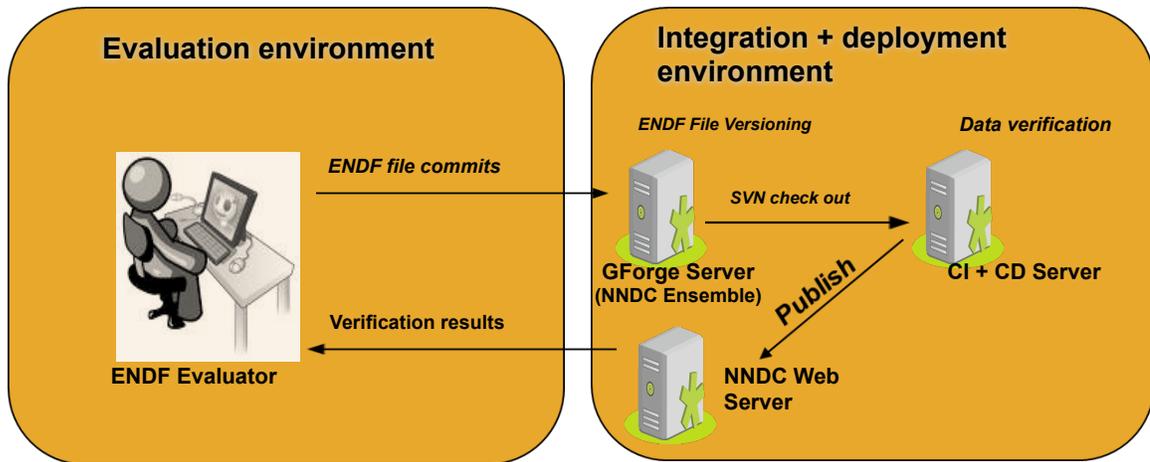


Fig. 1: Overview of the ADVANCE system as of version v0.7. This organization of version control and final publication should be maintained in the final product.

2. **Open source.** Many other members of the nuclear data community are interested in using ADVANCE. Therefore, the system must be released under an open license to foster these potential collaborations. As much of the code-base as possible should be included in the release. No export controlled codes or codes with otherwise limited distribution can be included. All authors of included codes should be consulted to obtain permission for inclusion in the ADVANCE code base.
3. **Documented.** The package must include good documentation. This includes, but is not limited to:
 1. *READMEs*, to describe installation and general use
 2. *Tutorial*, to enable a “quick-start” for new users
 3. *User guide*, to provide a detailed reference for users
 4. *Developer API*, to enable users to become collaborators and help improve the code base

5. *FAQ*, to collect frequently asked questions about ADVANCE.
 6. *Project website*, likely hosted on the NNDC GForge server, to collect and post these and other relevant documents.
4. **Tested.** ADVANCE must have its own ADVANCE site and a collection of unit tests that are run at regular intervals on the code system. These tests must also be runnable from the command line so that new users can run the tests to ensure a successful installation.
 5. **Self contained.** The final source code package must be as stand-alone as possible. Dependencies on external packages complicate maintenance for developers and installation for users. For all included packages, we require
 1. *All source code*, to enable an open source release
 2. *All inputs*, to enable a “quick-start” for new users
 3. *All documentation*, to provide a detailed reference for users. Some codes we use are legacy codes and the form and level of documentation may not meet modern standards
 4. *Unit tests*, runnable from the command line, so that users can be assured of a successful installation.
 5. *Must build/install cleanly*, to enable a “quick-start” for new users
 6. *A version number*, so that code issues can be properly attributed with the version of the code in question
 7. *A Point Of Contact (POC), with an email address*, so that code issues can be properly routed to responsible individuals

1.b. Architecture requirements

1. **Master-Slave Architecture.** Builds should be controlled by a central Build Master and the actual work distributed to Build Slaves. The slaves may be processes on the main build master machine or they may be local or remote computing resources. This architecture allows us to delegate build tasks to other machines, increasing the number of tests that may be performed on build targets. This architecture is illustrated in Fig. 2.

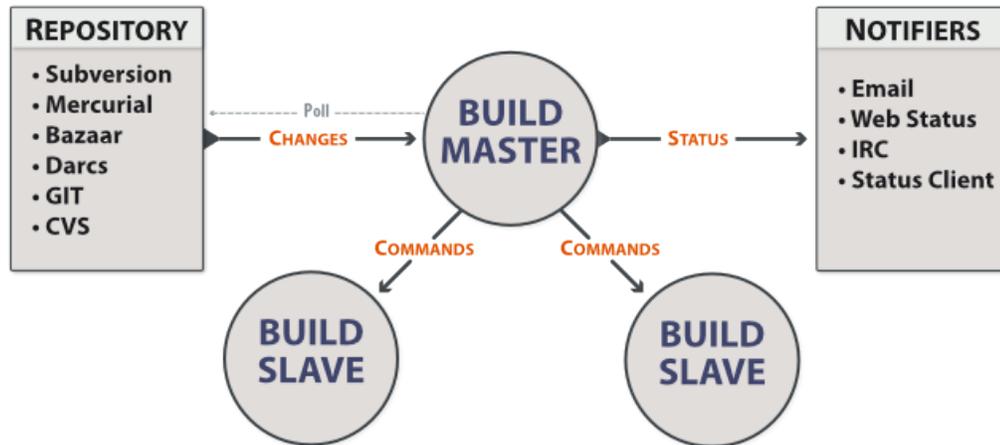


Fig. 2. The proposed arrangement of the Build Master and Build Slave nodes. (Taken from BuildBot project documentation).

2. **Build Triggers.** It should be possible to trigger a build in a variety of ways, including:
 1. *Command line*
 2. *Web interface*, in most off-the-shelf packages, this takes the form of a Status Dashboard application
 3. *Push request from an external code repository*
 4. *Poll an external repository and build if change occurred*
 5. *On a set schedule*, perhaps using a Unix cron-job
3. **Build Masters.** Build Masters must not only be able to communicate to the Build Slaves, but also to the deployment platform. This is most likely a webserver capable of serving up static build reports and build artifacts, which are rsync'ed from the Build Master. The Build Master may also contain a built-in webserver and Status Dashboard. Finally, the Build Master controls any Try/Accept functionality defined for the ADVANCE system (see Section 1.d).
4. **Self-contained Build Slaves.** Build Slaves should have all executable code on-node that they will need to run the requested jobs. They should have access to source code repositories to retrieve data needed for their build tasks.
5. **Build Slaves may delegate.** Build Slaves may be connected to clusters of their own and may have ability to launch jobs on that cluster using cluster-specific job controls. This capability will be essential for benchmarking with some of the more complicated benchmark models (e.g. LLNL Pulsed Spheres and Wyman Spheres).

1.c. Cyber security

The ADVANCE system must be compliant with all DOE and laboratory specific cyber security requirements. As the requirements may vary from somewhat from laboratory to laboratory, we must lay out some required capabilities that will enable use to meet these varied requirements.

WHAT ARE THEY? DO WE HAVE REFERENCES? A SUMMARY?

1. **Build Master security.** The Build Master controls the build and the jobs submitted to Build Slaves. Therefore it will likely require complicated configuration. Additionally, we want flexibility in both master and slave configurations to support many different working models including triggering jobs from the command line, reachable by ssh, or email or a web interface. Therefore we require:
 1. *A read-only dashboard for casual users.*
 2. *Password protected access.* Controls that force rebuilds and controls that allow a user to create new build commands must be password protected.
 3. *Definable roles.* Only some users can force builds and others may add more build commands.
 4. *No username/passwords should be stored in clear text.*

2. **Build Slave security.** Build Slaves should be able to reside behind a firewall. Only the Build Master should be able to submit jobs to Build Slaves unless one logs onto the Build Slave node directly and invokes commands from the command line. The Build Slaves should be able to return build results to the Build Master. The Build Master and Slaves must communicate through ssh. This arrangement is illustrated in Fig. 3.

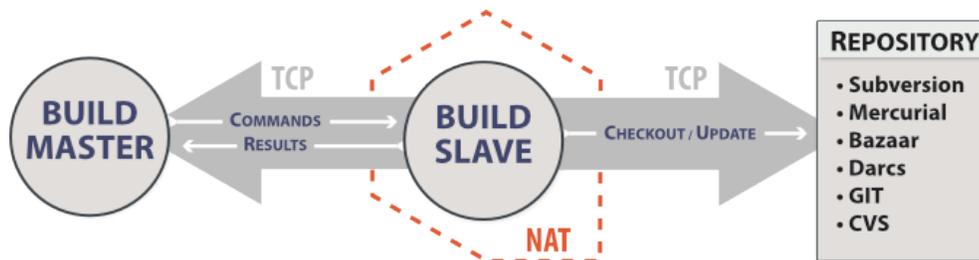


Fig. 3. The communications of the Build Slave with the Build Master and source control system. The Build Slave may live behind a firewall so long as ssh can tunnel through. (Taken from the BuildBot project documentation).

1.d. General functionality

There are many general purpose modules and features that must be included in ADVANCE.

1. **Try/Accept functionality.** The ability to upload a change set and rebuild all dependencies, then report to the requester. This allows a requestor to try out a change before actually committing it to the final repository. We must be able to password protect this functionality.
2. **Notifications.**
 1. *Real Simple Syndication (RSS) feeds*, for all levels of reporting
 2. *Email notification*, to previous committer upon completion of all relevant jobs
 3. *Build status page*
3. **2D plot requirements.** Most projects require some plotting capability for assembling useful status reports.
 1. *Easy to understand legends and axes labels*
 2. *Dynamic* (if possible), would have to use HTML5 canvas, otherwise using the PNG image format
4. **Report generation.**
 1. *Side-by-side HTML differences*
 2. *Email generator capability*
 3. *File parser*, help for legacy code outputs
 4. *Unit test reports*, triggered on successful build
 5. *Integration/benchmark reports*, triggered on successful build, but possibly only run a maximum of once per day
 6. *Links for build artifacts* (e.g. installation tarballs, jar files, etc.)
 7. *Links to relevant trackers*, likely in GForge
 8. *Ability to make comments on build target*
 9. *CMS and/or templating system*, for simpler report generation
5. **Look and feel.**
 1. *Maintainable CSS and Javascript*
 2. *Self-contained.* Styles/codes must be distributed with ADVANCE.
 3. *Consistent theme/style.* The style must not be ugly.

2. Nuclear reaction data QA requirements

The original purpose of ADVANCE was for nuclear data quality assurance. For this purpose, there are many existing test codes and a few documented standards that any evaluation must satisfy. Here we document the codes and requirements. Additionally, through interaction with our user base, we have determined what reports our users require. In this section we detail those as well.

2.a. Test codes

The nuclear data community has developed a number of codes for data quality assurance and data processing. These codes must be integrated into the system and in a form that the user can use simply. In addition, the outputs that these codes produce must be made understandable. The codes here must run quick enough that they can be used “in-line” that is, as soon as a changeset is detected by the CI/CD system.

1. **All common open source processing codes for ENDF files**
 1. *NNDC checking codes* (CHECKR, FIZCON, PSYCHE, etc.) [NNDC]
 2. *PREPRO*, Red Cullen’s preprocessing and checking codes (e.g. LINEAR, RECENT, SIGMA1, SIXPACK) [PREPRO]
 3. *Fudge-4.0*, LLNL’s new processing and data manipulation system [Fudge].
 4. *Others* (see Refs. [ACELST, ENDF2HTM], possibly PyNE [PyNE])
2. **Facilities for adding commonly used, but not open source, ENDF processing codes**
 1. *NJOY99/NJOY2012*, LANL’s processing system [NJOY99, NJOY2012]. Processes data for use in MCNP [MCNP6].
 2. *AMPX and PUFF*, ORNL’s processing system [AMPX]. Processes data for use in the SCALE package [SCALE].
 3. *CALENDF*, an CEA (France) processing code [CALENDF].
3. **Code maintenance**
 1. *Instructions/examples for adding more codes.*
 2. *Ability to generate an email to a code’s POC to report issues.*
 3. *Project pages and issue trackers, if available for codes.*
4. **Code output reports**
 1. *Pass/Fail/Warning notification*, usually in the form of “blinking lights”
 2. *Upon error, provide a link to a runlog*
 3. *Organized and interpreted list of notable messages*, including errors, warnings and failures
 4. *Links to all produced artifacts*

2.b. Quality assurance metrics

Although there is not much in the way of formal QA documentation for ENDF, the ENDF library is thoroughly tested. That said, data tested with ADVANCE must comply with all documented QA requirements.

1. Covariance reports

1. *Statement of consistency with requirements in Ref. [CovQA]*
2. *Cross section plots, with uncertainties and cross comparisons with EXFOR data*
3. *Plots of covariance, relative covariance or correlation matrices*
4. *Statement or plots showing uncertainties relative to stated uncertainty bounds stated in Ref. [CovQA]*

2. ACE file reports

1. *Statement of consistency with requirements in Ref. [ACEQA]*
2. *Cross section plots, including cross comparisons with EXFOR data, the original ENDF file and results from other processing codes, at 0 and 273°K*

The screenshot displays the main library overview report of the ENDF project from ADVANCE v0.7. The interface includes a navigation bar at the top with tabs for OVERVIEW, ENDF PROJECT, ADVANCE, and NNDC. The main header reads "ADVANCE: The ENDF Continuous Integration System" and features the Brookhaven National Laboratory logo. The central content area is titled "ENDF B-VII dev" and "ENDF/B Development", describing it as "The development version of the Evaluated Nuclear Data File (ENDF/B)". A "Latest Updates" section lists three recent reports: "sublib_release_notes: neutrons", "sublib_html: neutrons", and another "sublib_release_notes: neutrons" report, all dated 2013-04-30. Below this, a "Neutrons sublibraries" section is visible, with tabs for Neutrons, Decay, Charged particles, Photonuclear, and Atomic. Four sublibrary thumbnails are shown: "Neutrons Sublibrary" (a circular diagram with isotopes), "Neutron-Induced Fission Yields Sublibrary" (a diagram of a fission reaction), "Standards Sublibrary" (a photograph of a digital scale), and "Thermal Neutron Scattering Sublibrary" (a diagram of a neutron scattering experiment).

Fig. 4. Main library overview report of the ENDF project from ADVANCE v0.7.

2.c. Reports

In ADVANCE v0.7 [ADVANCE], there are different levels of reporting. Users find these different levels provide useful navigation and organization. Figs. 4 and 5 illustrate this. We would like to maintain this hierarchy including library wide, per sublibrary and per evaluation in sublibrary collections of reports.

1. Sublibrary wide report

1. *Simple navigation with periodic table.* See Fig. 5 for an example.
2. *Pass/fail summary* for all parts of sublibrary
3. *Current activity* on all parts of sublibrary
4. *Archive file of processed sublibrary, including checksums* (e.g. ACE files and xsdir file in a zipfile), ready for testing in an application
5. *Tracker links*

2. Reports for all types of evaluations

1. *Interpreted ENDF file overview*
2. *On the fly rolling elemental evaluations*
3. *Tracker links for sublibrary*
4. *Summary of current activity on an evaluation*

3. Specific reports for reaction evaluations, see Fig. 5 and 7

1. *Covariance reports*, as discussed in requirement 2.b.1
2. *ACE file reports*, as discussed in requirement 2.b.2
3. *Energy balance/KERMA report*, to demonstrate that reaction data is consistent with bounds imposed by energy conservation
4. *Forward momentum balance*, to demonstrate that reaction data is consistent with bounds imposed by momentum conservation
5. *Benchmarking report*, as discussed in requirement 2.d.8
6. *Plots with EXFOR data*,
 - d. Cross sections, as discussed in requirements 2.b.1 and 2.b.2
 - e. Average forward scattering angle (mubar)
 - f. Average fission neutron multiplicity (nubar)
7. *Integral quantities*, with uncertainties if available
 - a. Maxwellian Average Cross Sections (MACS)
 - b. Resonance Integral (RI)
 - c. Wescott Factor
 - d. 14 MeV point
 - e. Comparisons to Atlas values, values derived from other libraries, including Ref. [KaDoNiS]
8. *For neutrons, an additional resonance report*
 - a. RRR, URR, high-energy region matching
 - b. PSYCHE's tests [NNDC]
 - c. Resolved resonance statistics, including mean level spacing, the Δ_3 statistic, and S and P-wave strength functions

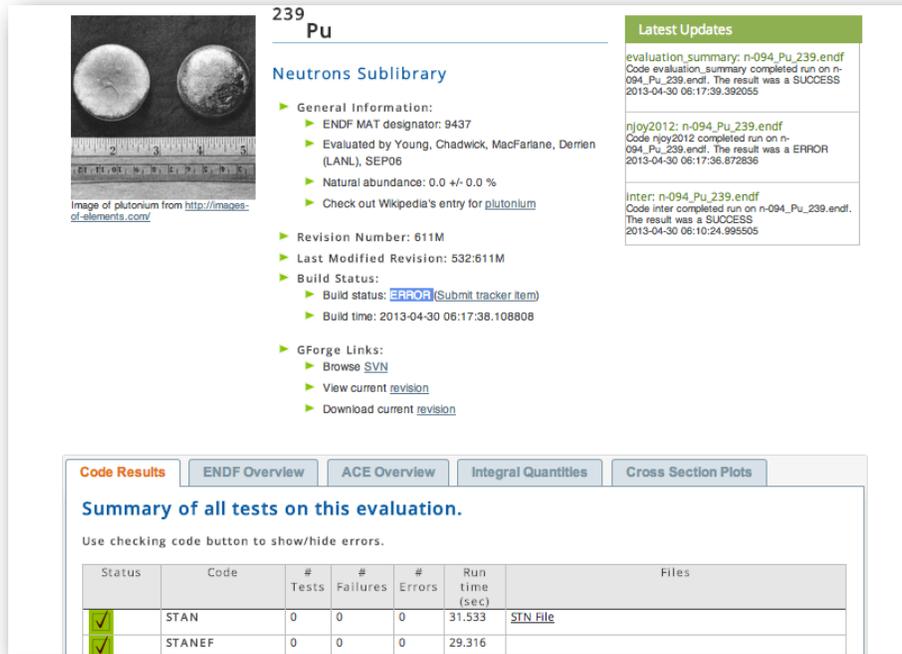


Fig. 6. A sample evaluation webpage from ADVANCE v0.7. The tabs in the page show different reports.

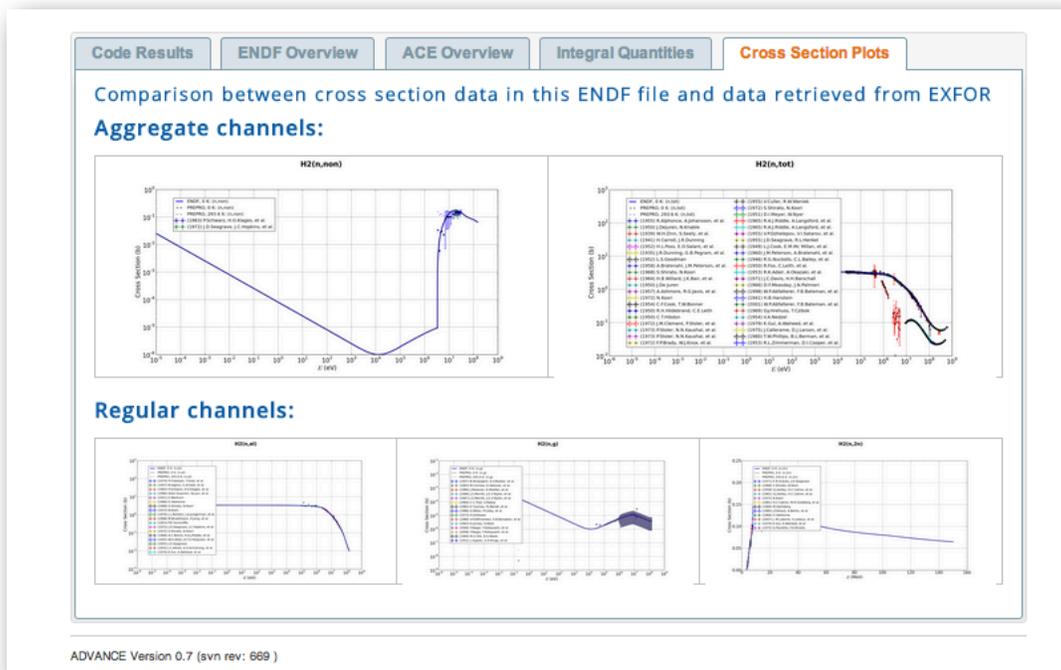


Fig. 7. Sample plots from ADVANCE v0.7's cross section plot report.

2.d. Benchmarking

Benchmarking is a critical part of our data testing regimen but the benchmark tests are often complex and take a long time to run. They are not amenable to “on-the-fly” testing such as envisioned for the usual tests, but must be scheduled somehow. The requirements listed below are heavily based on the capabilities of Ref. [cnp_test_suite] from LLNL.

1. **Job handling**
 1. *Run on Build Slave only*
 2. *Can set schedule*
 3. *Can delegate to cluster*
 4. *Build Slave may be a remote resource.* It may not even be on the same domain as the Build Master.

2. **On-node database.** Each Build Slave must contain a database of all benchmark tests it is capable of running.
 1. *Searchable by ICSBEP id [ICSBEP]* (or other id from other benchmark libraries)
 2. *Contain the input decks* for the codes supported by the Build Slave
 3. *Optional searchable string tags* (“my-favorite”, “PMF-quick-test”, “lots-of-iron”)

3. **Target substitution.** The benchmarking system must support on-the-fly isotope substitution to swap natural to isotopic or monoisotopic to natural, based on library contents.

4. **Workflow of a Build Slave.**
 1. Query database for tests affected by current changeset
 2. Do isotope substitution if needed
 3. Assemble job(s)
 4. Assemble ancillary data needed by codes
 5. Possibly submit jobs to a batch queue controlled by the Build Slave
 6. Build benchmark report after job(s) finish

5. **Classes of benchmark tests.**
 1. *Random reflectors*, a simple “smoke test” where a reflector of the chosen material is placed around a simple configuration such as Godiva and a static eigenvalue calculation is performed. The k_{eff} is irrelevant, we are only interested in whether the material crashes the code in question. The test should run fast.
 2. *Broomstick*, a simple “smoke test” where we pulse a beam of neutrons or charged particles through a “broom stick” of the material to be tested.

3. *R. Mosteller's MCNP suite [Mosteller]*, a series of static criticality tests from the ICSBEP handbook.
4. *COG suite*, the test suite included with the COG code system [COGSuite].
5. *Jezebel & Godiva*, the “simplest” and “best characterized” critical assemblies
6. *Activation tests*, foils activated by irradiation in the neutron field of a critical assembly. These are very useful for testing specific reactions.
7. *WHAT TESTS FOR DECAY DATA???*

6. Application code support.

1. *Codes implemented at BNL*
 - a. MCNP6, LANL Monte-Carlo transport code [MCNP6]
 - b. TWOTRAN, a legacy ORNL deterministic transport code [TWOTRAN]
 - c. ANISN, a legacy ORNL deterministic transport code [ANISN]
 - d. COG, LLNL Monte-Carlo transport code [COG]
2. *Support for other popular codes*
 - a. Mercury, LLNL Monte-Carlo transport code [Mercury]
 - b. AMTRAN, LLNL deterministic transport code [AMTRAN]
 - c. ARDRA, LLNL deterministic transport code [ARDRA]
 - d. PARTISN, LANL deterministic transport code [PARTISN]
 - e. GEANT4, open source Monte-Carlo transport code hosted by CERN [GEANT4]
 - f. FLUKA, ICFN (Italy) Monte-Carlo transport code [FLUKA]
 - g. SCALE, ORNL Monte-Carlo and deterministic transport code system including other codes such as ORIGEN, TSUNAMI and KENO [SCALE]
 - h. SERPENT, European Monte-Carlo transport code [SERPENT]
 - i. TRIPOLI, European Monte-Carlo transport code [TRIPOLI]
 - j. CINDER, LANL burn-up code [CINDER]

7. **Data sensitivities.** If a code can generate sensitivities automatically, we need a sensitivity report. Both MCNP6 and TSUNAMI are capable of this.

8. Benchmark reports.

1. *All results of same type classified together*
2. *Sort/search on material, tags, ICSBEP identification, spectral index*
3. *1d data (e.g. k_{eff}): absolute and Calculation/Experiment as table and plot*
4. *2d data (spectra): plot only*
5. *Observables for critical assemblies:*
 - a. k_{eff} and β_{eff}
 - b. spectrum (per E_{out} or Time Of Flight (TOF))
 - c. leakage fraction (e.g. Above Thermal Fraction (ATF))
 - d. spectrum weighted yields (e.g. foils)
6. *Observables for accelerator driven systems (e.g. LLNL Pulsed Spheres and Wyman Spheres and other d-t sources)*
 - a. spectrum (per E_{out} or TOF)
 - b. spectrum weighted yields
7. *WHAT OBSERVABLES FOR DECAY BENCHMARKS???*

3. Product Comparison: BuildBot vs. RunDeck vs. Jenkins

In this section, we perform a product comparison between three of the most popular CI/CD systems for possible adoption into the ADVANCE code system: BuildBot, RunDeck and Jenkins.

About BuildBot

An open source, pure Python project. It is popular: it is used in both the Mozilla and Google Chrome projects for their CI/CD needs. BuildBot needs several third party components all of which are open source and readily accessible. [BuildBot]

About RunDeck

Another open source project, written in Java. RunDeck is not a CI/CD system, but rather a distributed command execution framework. Therefore it is very flexible. RunDeck is derived from DTO Solutions' ControlTier framework which has been discontinued. [RunDeck]

About Jenkins

An open source fork of the Hudson CI/CD project. Jenkins is "turn-key" and very easy to start using. However, it is very software-project focused so adapting it to a data project may be complex. Jenkins is quite popular, but not as popular as BuildBot. [Jenkins]

	BuildBot	RunDeck	Jenkins
URL	http://buildbot.net	http://rundeck.org	http://jenkins-ci.org
Open source license	GNU Public License	Apache Software License	MIT License
Continuous Deployment	Yes	No	Yes, with plugin
Available source	Yes	Yes	Yes
Locally available documentation	Yes	Yes, must install from source	Yes, must install from source
Built in webserver	Yes (uses Twisted)	Yes	Yes (not obvious how to turn it off)
Try/Accept functionality	Yes	No	No

	BuildBot	RunDeck	Jenkins
Notifications	Yes (email, RSS, IRC)	Yes (RSS)	Yes (email, RSS, IRC)
GForge compatibility	SVN polling	SVN polling	SVN polling; "Blinking lights" plugin
CLI	Yes	Yes, requires separate package	Yes, requires separate package
Unit testing	Yes	No	Yes
Templating/Report generation	Yes (uses Jinja2)	No	No
Distributed execution (Master/Slave architecture)	Yes	Yes	Yes
Support available	Mailing list: http://lists.sourceforge.net/lists/listinfo/buildbot-devel ; Bug tracker: http://buildbot.net/trac/newticket , IRC: #buildbot channel on Freenode	Mailing list: http://groups.google.com/group/rundeck-discuss ; IRC: irc://irc.freenode.net/rundeck; man pages	Several commercial support options available from 3rd party vendors; Mailing lists: jenkinsci-users@googlegroups.com
Language	Python	Java	Java

Recommendation.

Our recommendation is to use BuildBot. It is popular and open source, meaning that the project is likely to survive for quite some time. If the project does collapse, we will at least have the full source code and documentation and can continue with our own internal version. BuildBot is feature rich, including the try/accept functionality, which is absent from other packages. Finally, BuildBot is written in Python so it fits well with the rest of the ADVANCE code base which leverages Python's subprocess controls and uses Fudge for ADVANCE's plotting.

References

- [ACELST] A. Trkov, ACELST, personal communication.
- [ACEQA] A. Trkov, IAEA Report INDC(SEC)-0107 Guidelines for Nuclear Data Verification and Validation Aug 2005 <http://www-nds.iaea.org/publications/indc/indc-sec-0107/>
- [ADVANCE] ADVANCE v0.7, <https://ndclx4.bnl.gov/gf/project/checkendf/>.
- [AMPX] <http://www.ornl.gov/sci/scale/overview/ampx.htm>.
- [AMTRAN] <https://e-reports-ext.llnl.gov/pdf/242941.pdf>, <https://wci.llnl.gov/codes/mercury/media/pdf/Amtran.pdf>
- [ANISN] R. Douglas O'Dell and Raymond E. Alcouffe: Transport Calculations for Nuclear Analyses: Theory and Guidelines for Effective Use of Transport Codes LA-10983-MS and UC-32 (September 1987).
- [ARDRA] ARDRA, a scalable parallel code system to perform neutron and radiation transport calculations, http://computation.llnl.gov/casc/Ardra/tech_brochure98/ardra_flier.html.
- [Atlas] Said Mughabghab, Atlas of Neutron Resonance Parameters and Thermal Cross Sections. Z=1-100 Elsevier ISBN: 978-0-444-52035-7 (2006)
- [BuildBot] BuildBot, continuous integration framework, <http://buildbot.net/index.html>
- [CALENDF] J-Ch Sublet, P Ribon, M Coste-Delclaux, CALENDF-2010: User Manual Rapport CEA-R-6277, ISSN 0429-3460, 2011.
- [CINDER] W. B. Wilson, S. T. Cowell, T. R. England, A. C. Hayes & P. Moller: A Manual for CINDER'90 Version 07.4 Codes and Data, LA-UR-07-8412 (December 2007, Version 07.4.2 updated March 2008)[cnp_test_suite]
- [COG] COG: A High Fidelity Multi-Particle Transport Code, <http://cog.llnl.gov/>.
- [COGSuite] D. Heinrichs, COG test suite, personal communication.
- [CovQA] D. Smith, "Quality Assurance Requirements for ENDF/B-VII.1 Covariances"
- [ENDF] M. B. Chadwick, M. Herman, P. Oblozinsky, et al., "ENDF/B-VII.1 nuclear data for science and technology: Cross sections, covariances, fission product yields and decay data", Nuclear Data Sheets, 112(12):2887-2996 (2011).
- [ENDF2HTM] R. MacFarlane, ENDF, personal communication.
- [FLUKA] FLUKA, Monte Carlo simulation package, <http://www.fluka.org/fluka.php>.
- [Fudge-4.0] C.M. Mattoon, B.R. Beck, N.R. Patel, N.C. Summers, G.W. Hedstrom, D.A. Brown, Generalized Nuclear Data: A New Structure (with Supporting Infrastructure) for Handling Nuclear Data, Nuclear Data Sheets, Volume 113, Issue 12, December 2012, Pages 3145-3171, ISSN 0090-3752, <http://dx.doi.org/10.1016/j.nds.2012.11.008>. (<http://www.sciencedirect.com/science/article/pii/S0090375212000944>) <https://ndclx4.bnl.gov/gf/project/gnd/>
- [GEANT4] Nuclear Instruments and Methods in Physics Research [A 506 \(2003\) 250-303](#), and IEEE Transactions on Nuclear Science [53 No. 1 \(2006\) 270-278](#). <http://geant4.cern.ch/>.
- [GForge] GForge, collaboration environment, <http://gforgegroup.com/>.
- [git] git, distributed version control system, <http://git-scm.com/>.
- [ICSBEP] <http://icsbep.inel.gov/>; <http://www.oecd-nea.org/science/wpncs/icsbep/>

- [Jenkins] Jenkins, An extendable open source continuous integration server, <http://jenkins-ci.org/>.
- [KaDoNiS] KADoNiS v0.3 - The third update of the "Karlsruhe Astrophysical Database of Nucleosynthesis in Stars" I. Dillmann, R. Plag, F. Käppeler, T. Rauscher Submitted as proceeding of the workshop "EFNUDAT Fast Neutrons - scientific workshop on neutron measurements, theory & applications" held on April 28-30 2009 at Geel, Belgium. <http://www.kadonis.org/>.
- [MCNP6] MCNP6, a general Monte Carlo N-particle transport code, <http://mcnp.lanl.gov/>, LANL report LA-UR-11-07082.
- [Mercury] Mercury, a modern Monte Carlo particle transport code, <https://wci.llnl.gov/codes/mercury/index.html>.
- [Mosteller] R. Mosteller, personal communication.
- [NJOY99] R. MacFarlane, NJOY99, <http://t2.lanl.gov/nis/codes/njoy99/index.html>
- [NJOY2012] A.H. Kahler, R. MacFarlane, NJOY2012, <http://t2.lanl.gov/nis/codes/NJOY12/index.html>; User Manual LA-UR-12-27079.
- [NNDC] NNDC checking codes (STAN, STANEF, CHECKR, FIZCON, PSYCHE)
- [ORIGEN] Included in ORNL/TM-2005/39, SCALE Version 6.1 (June 2011)
- [PARTISN] R. E. Alcouffe, R. S. Baker, J. A. Dahl, S.A. Turner, and Robert Ward: PARTISN: A Time-Dependent, Parallel Neutral Particle Transport Code System, LA-UR-08-07258 (Revised Nov. 2008); R. E. Alcouffe, R. S. Baker, F. W. Brinkley, D. R. Marr, R. D. O'Dell, and W. F. Walters: DANTSYS: A Diffusion Accelerated Neutral Particle Code System, LA-12969-M (1995)
- [PREPRO] D. Cullen, PREPRO 2012, <http://www-nds.iaea.org/public/endl/prepro/>.
- [PyNE] A. Scopatz, S. Johnson, P. Romano, et al. PyNE: The Nuclear Engineering Toolkit, <http://pynesim.org/>.
- [RunDeck] RunDeck, a remote command dispatch framework, <http://rundeck.org/>.
- [SCALE] ORNL/TM-2005/39, Version 6.1 (June 2011)
- [SERPENT] SERPENT, a continuous-energy Monte Carlo reactor physics burnup calculation code, <http://montecarlo.vtt.fi>.
- [subversion] <http://subversion.apache.org/>; B. Collins-Sussman, B.W. Fitzpatrick, C.M. Pilato, Version Control with Subversion, O'Reilly Media (2011), ISBN 978-0-596-51033-6, available at <http://svnbook.red-bean.com/>.
- [TRIPOLI] E. Brun, F. Damian, E. Dumonteil, F.-X. Hugot Y.-K. Lee, F. Malvagi, A. Mazzolo, O. Petit, J.-C. Trama, T. Visonneau, A. Zoia: TRIPOLI-4 version 8 User Guide, CEA-R-6316, Feb. 2013.
- [TWOTRAN] K. D. Lathrop, and F. W. Brinkley; TWOTRAN-II: An Interfaced, Exportable Version of the TWOTRAN Code for Two-Dimensional Transport. LA-4848-MS, UC-32 (July 1973). K. D. Lathrop and F. W. Brinkley, Theory and Use of the General-Geometry TWOTRAN Program, LA-4432, April 1970. K. D. Lathrop, F. W. Brinkley, and P. Rood, Theory and Use of the Spherical Harmonics, First Collision Source, and Variable Weight Versions of the TWOTRAN Transport Program, LA-4600, March 1972. R. Douglas O'Dell and Raymond E. Alcouffe: Transport Calculations for Nuclear Analyses: Theory and Guidelines for Effective Use of Transport Codes LA-10983-MS and UC-32 (September 1987)