



# ADVANCE Discussion

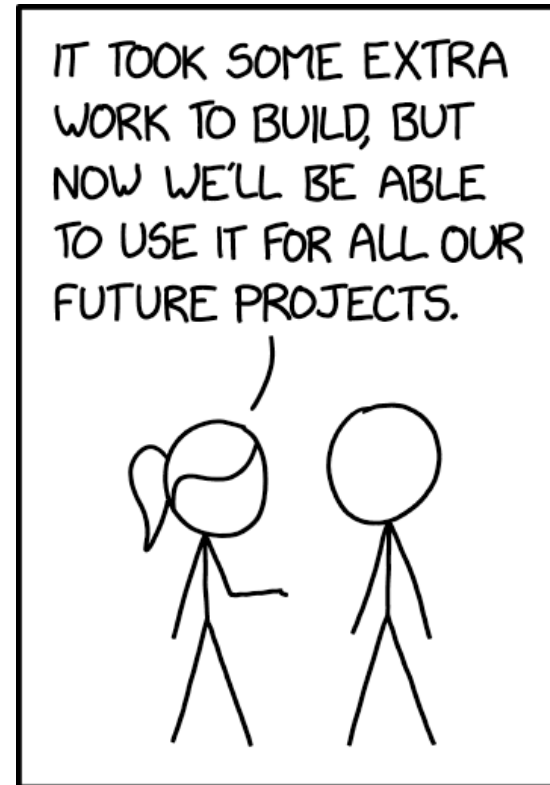
David Brown  
National Nuclear Data Center

25 April 2023

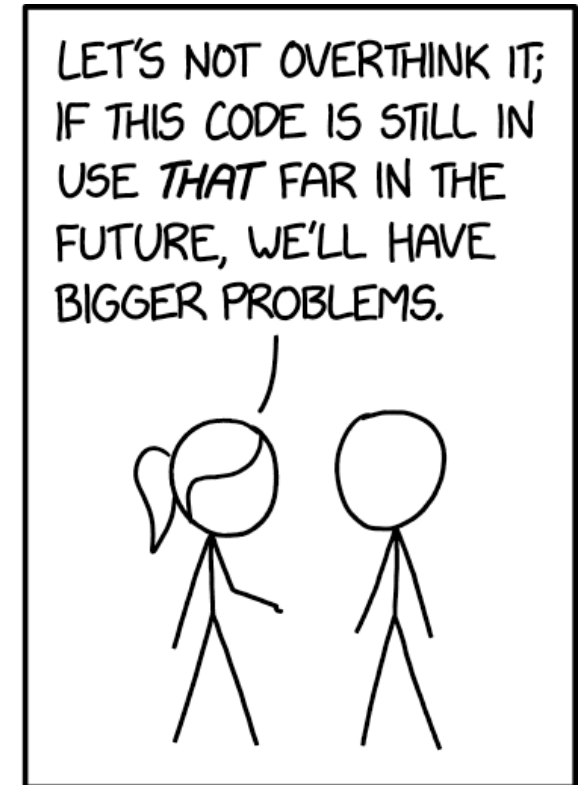


# ADVANCE is the ENDF continuous integration system

- Stood up as quick-n-dirty hack to get ENDF/B-VII.1 out the door  
... in 2011
- Faithfully (if at times erratically) serving CSEWG since
- All ENDF files ran through variety of physics & format checks as well as processing codes



HOW TO ENSURE YOUR CODE IS NEVER REUSED



HOW TO ENSURE YOUR CODE LIVES FOREVER

# Treating Data Like Software: A Case for Production Quality Data

Jennifer M. Schopf

Woods Hole Oceanographic Institution

Woods Hole, MA 02543

(Currently at the National Science Foundation, GEO/OAD)

jschopf@whoi.edu

Jennifer M. Schopf. 2012.  
Treating data like software: a  
case for production quality data.

In Proceedings of the 12th  
ACM/IEEE-CS joint conference  
on Digital Libraries (JCDL '12).

Association for Computing  
Machinery, New York, NY, USA,  
153–156.

DOI:<https://doi.org/10.1145/2232817.2232846>

## ABSTRACT

In this short paper, we describe the production data approach to data curation. We argue that by treating data in a similar fashion to how we build production software, that data will be more readily accessible and available for broad re-use. We should be treating data as an ongoing process. This includes considering third-party contributions; planning for cyclical releases; bug fixes, tracking, and versioning; and issuing licensing and citation information with each release.

## Categories and Subject Descriptors

E.5.3 [Data]: Files - *Organization, Structure*; E.4.3 [Data]: Coding and information theory - *Formal models of communication*; H.1 [Information Systems] - Models and principles

## General Terms

Management, Documentation, Design, Verification

## Keywords

Best practices, Cyclical development and release,  
Production data

available digitally, the ability to find and access data is increasingly difficult.

In order to address the need for better data preservation and access, we propose that data sets should be managed in a similar fashion to how we maintain production quality software. These *production data sets* are not simply published once, but go through a cyclical process of development, verification, deployment, support, analysis, and then development again. Attention is given to ensuring the data is understandable, useful, and updated over time, the same way software products need updating over time, even if the core functionality does not change.

This short paper gives a brief definition of what is meant by data in this context. It then addresses at a high level standard factors that are part of the development of (academic) production software, and describes how similar processes can be applied to enable data sets to have extended lifecycles and improved usability. A key premise is that if this approach can be integrated into common practice, it will result in a higher level of preservation and usability of data.

# Phase I testing automated for nearly 10 years

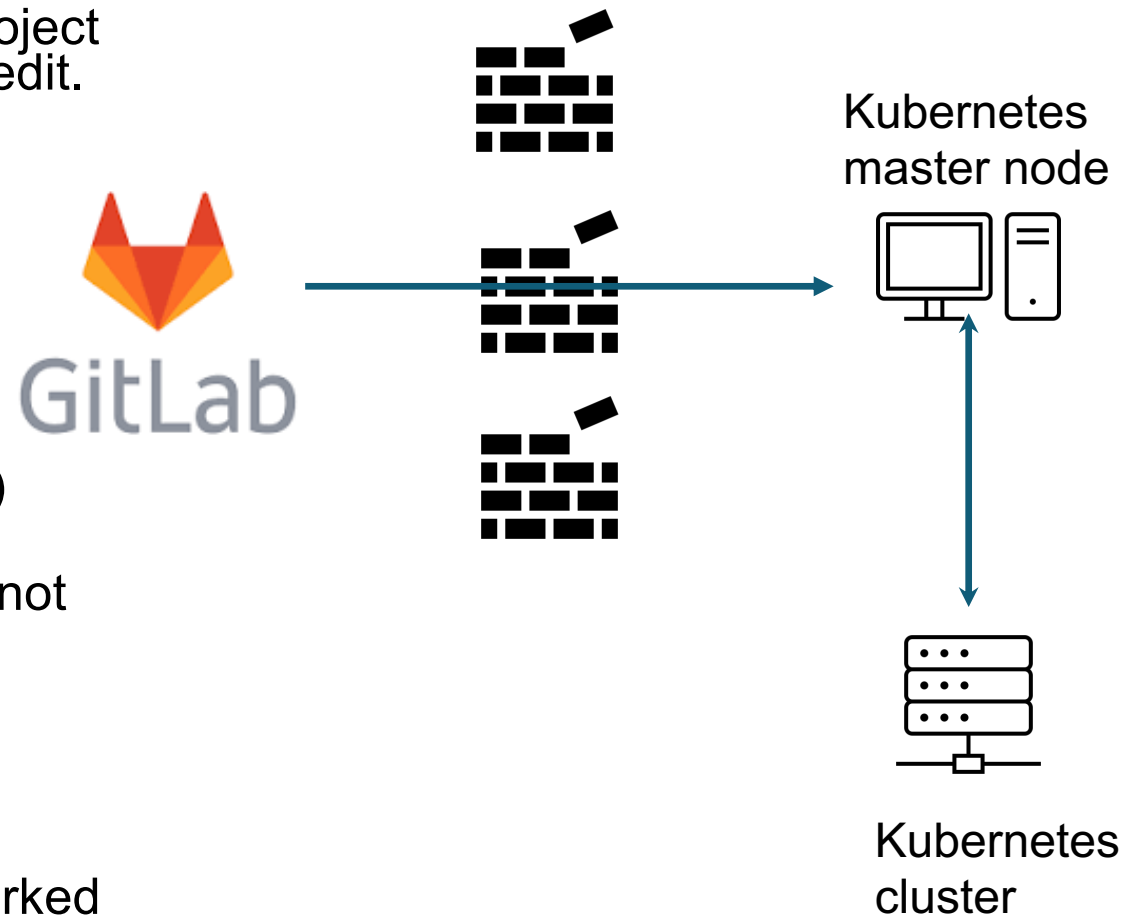
Can we automate reading build reports?  
What other tests can we automate?



Code	Test	pre-VII	Now
	File summary complete & correct	🤔	
STAN, STANEF, CHECKR, fudge	ENDF format compliance	🖥️	🖥️
FIZCON, fudge	Mathematical correctness (e.g. probabilities valid, covariances positive)	🖥️	🖥️
FIZCON, PSYCHE, fudge	Physical correctness (e.g. Q, thresholds, energy deposition/KERMA)	🖥️	🖥️
INTER, fudge (inter.py)	Compute & check integral metrics (e.g. RI, thermal cross sections, MACS)	🖥️	🖥️
fudge	Completeness (all outgoing particles, including gammas)	🤔	🖥️
ADVANCE	Comparisons to microscopic experimental data (EXFOR)	🤔	🖥️
	Assessment of application suitability (e.g. usable for fast reactors or spaceflight)	🤔	
	Reasonable (e.g. covariances, angular distributions)	🤔	
<b>fudge (grokres.py)</b>	<b>Resonance quality</b> (missing resonances? widths realistic?)		🖥️
PREPRO, fudge, NJOY (not SCALE yet)	Can process for user codes		🖥️
	Is state of the art? Is best we can do?	🤔	

# Major reworking of ADVANCE innards

- Gitlab's Ci/CD configuration controlled by per-project YAML file anyone who is part of the project can edit.
- To prevent accidental (or intentional?) troubles, must execute checking codes in container
- Gitlab, the Kubernetes master node (development2) and the Kubernetes worker node (ADVANCE2 server) are all behind the BNL FireWall
- Reverse proxy server (outside the BNL FireWall) secures access from the Internet to GitLab.
- Two configuration requirements Gitlab.com had not anticipated:
  - Our cluster must live behind firewall
  - Legacy checking codes need temp space
- Gitlab and BNL/ITD cyber security engineers worked with us to resolve issues.



# Simpler reports

- Reports will be per-commit, on any git branch (but probably restricted to review branches for now)
- They must be light weight, but not compromise content
- Solution:
  - Summary markdown, per code, replacing website
  - Any important build artifacts made by code (xsd, ace files, ...)
  - Pictures & lists of bugs in summary markdown

## Thanks to:

- Ramon Arcilla (BNL-NNDC) for fighting the fight with GitLab and
- Rebecca Coles (BNL-NNP) for creating the simplified reports

# Executes on every commit on every branch

The screenshot shows a CI/CD interface for a project named 'alphas'. The left sidebar has 'Pipelines' highlighted. The main area shows a table of pipeline runs:

Status	Pipeline	Triggerer	Stages	Actions
passed	Alpha files added #2550	main - 0fdbb956	Two green checkmarks	Download icon
passed	He-4 and Li-6 files added #2548	main - a03a1fdf	Two green checkmarks	Download icon
passed	Rest of alphas added #2545	main - 37bed775	Two green checkmarks	Download icon

Annotations include:

- A blue circle around the 'Pipelines' menu item in the sidebar.
- A blue circle around the 'Pipelines' tab in the top navigation.
- A blue circle around the 'passed' status for the first pipeline.
- A blue circle around the two green checkmarks in the 'Stages' column for the first pipeline.
- A blue circle around the download icon for the first pipeline.






Text annotations:

- 'It passed!' with an arrow pointing to the first pipeline's status.
- 'Multiple ways to find build artifacts in addition to main project page' with an arrow pointing to the download icon.

On the right, a ZIP file icon is labeled 'artifacts.zip'.

# Execution controlled by a YAML file

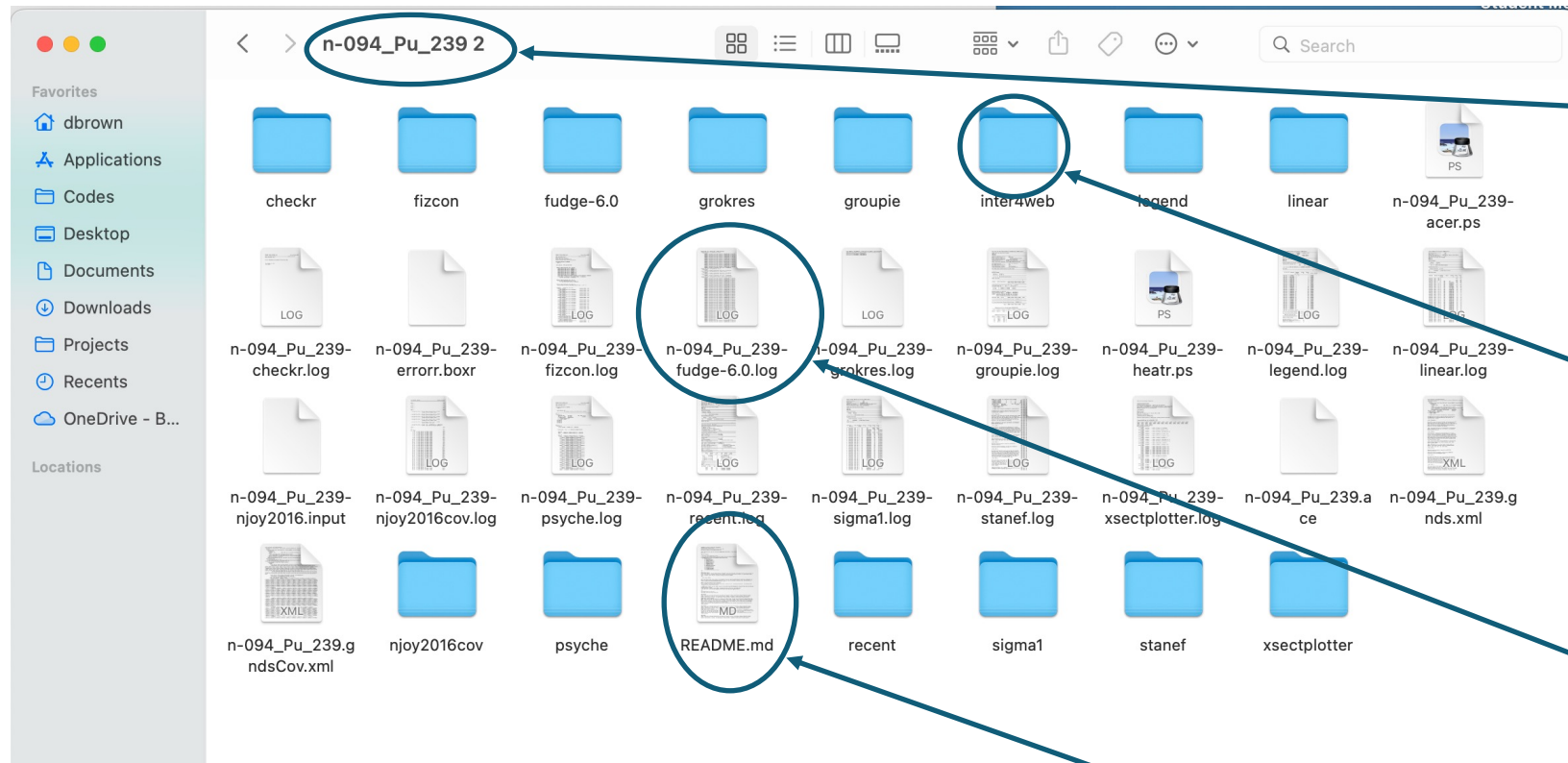
main ▾ alphas / .gitlab-ci.yml Find file Blame History Permalink

 **.gitlab-ci.yml**  2.18 KIB Edit in pipeline editor Lock Replace Delete   

```
1 variables:
2   OS_IMAGE: alpine:3.15.4
3   DOCKER_IMAGE: docker:19.03.12
4   #
5   # Set DOCKER_TLS_CERTDIR to "" to disable TLS use
6   # Otherwise, you get an error of 'client HTTP request to HTTPS server'
7   DOCKER_TLS_CERTDIR: ""
8   SHARED_PATH: /builds/${CI_PROJECT_PATH}/shared
9   ADVANCE_IMAGE: git.nndc.bnl.gov:5050/nndc/advance/advance-beta/advance:latest
10
11 stages:
12 - login
13 - verify
14
15 registry_login:
16   stage: login
17   image:
18     name: "${DOCKER_IMAGE}"
19   services:
20 - name: docker:19.03.12-dind
21   alias: docker
22   # THIS IS IMPORTANT!
23   command: ["--tls=false"]
24   script:
25 - export DOCKER_HOST=tcp://docker:2375
26 - echo ${DOCKER_HOST}
27 - docker login -u "${CI_REGISTRY_USER}" -p "${CI_REGISTRY_PASSWORD}" git.nndc.bnl.gov:5050
28   only:
29     changes:
30     - "*.endf"
31   except:
32     changes:
```



# An unpacked artifacts.zip file



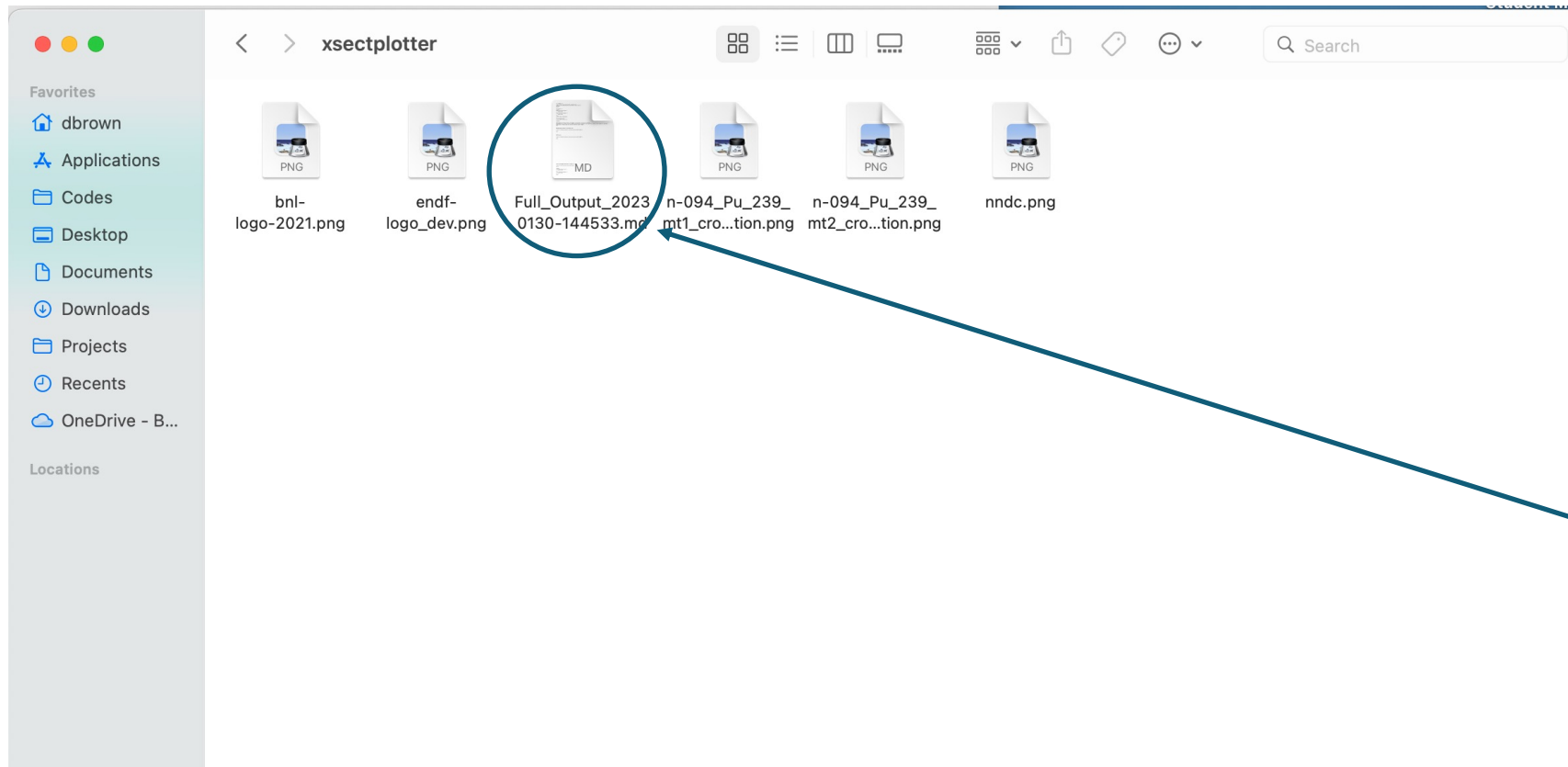
Each changed evaluation gets its own directory

Full build artifact for each code in own directory

stdout stream from code

Helpful instructions

# Sample directory




Open this. "Full\_output\_..."

# Output is in markdown and can be viewed in many text editors and web browsers

```
Full_Output_20230130-14... x
1 |!-- Header -->
2 
3 
4 <br><br>
5 <hr />
6
7 <!-- Style -->
8 <style>
9 table th:first-of-type {
10 width: 10%;
11 }
12 table th:nth-of-type(2) {
13 width: 90%;
14 }
15 body{
16 font-family: Monospace;
17 }
18 h1,h2,h3,h4,h5,h6{
19 font-family: Helvetica;
20 }
21 </style>
22
23 ### Comparison between data in this ENDF file and data retrieved from EXFOR
24 The EXFOR data is taken from the [EXFOR library](https://www-nds.iaea.org/exfor/). Note
25 - number of available EXFOR sets is large, they will not be listed in a plot legend.
26 <br>
27
28 ### Aggregate Channels Including Total
29
30 
31
32 <br>
33
34
35
36 ### Elastic
37
38 
39
40 <br>
41
42
43
```

Full\_Output\_2023013... x

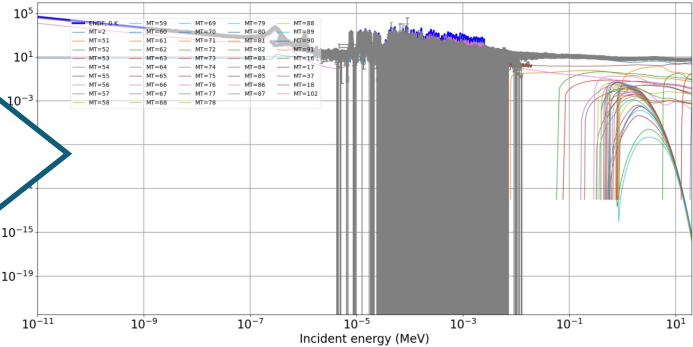


### Comparison between data in this ENDF file and data retrieved from EXFOR

The EXFOR data is taken from the [EXFOR library](https://www-nds.iaea.org/exfor/). Note: if the number of available EXFOR sets is large, they will not be listed in a plot legend.

#### Aggregate Channels Including Total

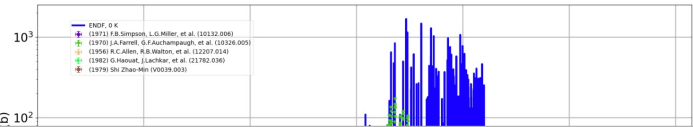
##### Pu239(n,tot)



atomi.io does this!

##### Elastic

##### Pu239(n,el)



Full\_Output\_20230130-144533.md 1:1

Screenshot

LF UTF-8 Markdown GitHub Git (0)

# Where are we now

- Runs on any commit to phase1  
(we can change it, but files take up a lot of space)
- We can adjust review forms to point to the correct artifact URL
- We hope (expect?) you all will have many comments about the build reports
  - Good comments: [dbrown@bnl.gov](mailto:dbrown@bnl.gov)
  - Bad comments: [gnobre@bnl.gov](mailto:gnobre@bnl.gov)

# What next?

- inter4web redo
- energy balance (we have PSYCHE, FUDGE and NJOY)
- spectra (gammas, vs. cap gam?)
- levels match ENSDF/RIPL (GRIN widget nearly ready)
- AMPX
- covariances (Denise has a wish-list)
- stronger, more elaborate, TSL checks (Chris & Ayman have wish-list)
- update resonances report with more robust CDF use for parameter extraction, explanation, and remove missing resonance tests (they are not ready for prime time)

If you belong to the ENDF family of projects, I can add you to  
ADVANCE and you can run it on your desktop