ESIP Software Guidance
Part 1
Soren Scott

ESIP Summer Meeting July 2016
A couple of definitions.

**Research code:** some code product that leads to some published outcome but is not a specific deliverable and usually not intended for reuse outside the project.

**Research software:** some code product that is intended for reuse outside the project (and is likely a specified deliverable).
Why software assessment?

We have the Technology Evaluation Framework (TEF) (Graybeal, 2016). It is criteria from the Software Sustainability Institute (and a few additions) applied to NASA’s Technology Readiness Levels.

It does what it says on the tin.
We want to move beyond formal assessment to education and mentoring for ESIP. The existing structure is not well-suited to this.

We started with restructuring and revisions for current developer practices if necessary.
A lot of things were said.
First.

We are covering a broad range of research dev activities.
Second.

PIs are the priority audience. They are between the funders (assessment) and the developers (of any description).
Third.

We want the document to be understandable to a broad audience.
Fourth.
The document is, itself, a means of education.
So we’re taking a step back from structured assessment criteria (today) to focus on putting together a broad set of principles and guidelines to address these notions.
What we’re doing today.
1. Scenarios
2. Specific guidelines discussions
3. Gap analysis – what are we missing
Ready?
Scenarios
Our diverse research community develops these outputs in diverse environments. Legacy systems, multiple funders, differing expectations per grant.
First, we want people to see how these guidelines apply to their work.

Second, we want to provide guidance for assessing these outputs across different situations.
Scenario Collection

https://titanpad.com/0z9RcazAJy
The Guidelines

http://goo.gl/w1YVF1
One last thing…
Language- and philosophy-agnostic. Avoid jargon but use real terms. Conceptual view. Point to industry practices if possible.
Sustainable Code
Interoperable
Usable
Documented
Secure
Sharable
Governed
Code as Research Object
Sustainable Code.

Clean, standardized code.
Versioned code.
Buildable/Installable.
Tested.
Interoperable

Data services

Community standards support

Semantic Services
Usable

Clear, understandable interface.
Software is performant.
Source code is documented.
Documentation supports adoption and reuse.
Software is documented.
Project is documented.
Secure
Sanitized source code.
Follows industry practices:
Web
Systems

Collaborative systems are secure.
Container security.
Sharable

Source code is licensed

Configurations included for automation.

Project name is discoverable.
Governed

Contribution policies are provided.
Development activities are transparent.
Code as Research Object
Publication and Citation
Preservation/Archiving
Credit
So what’s missing?
What do we need to say about how a piece of code or software works within the research group’s existing systems?
If you would like to be listed as a participant/contributor, check in the session with ImHere or add an annotation (with your name & organization) or let me know.
Thank you.