Software Evaluation: Concepts and Rubrics

Soren Scott
The Ronin Institute for Independent Scholarship
BESSIG February 22, 2016

A bit of history…

ESIP Products & Services committee has an incubator testbed

Last fall, with support from NASA, a pilot tech evaluation process was developed

The evaluation metrics were developed as a testbed project using the AIST Tech Readiness Levels and criteria developed by the Software Sustainability Institute

NASA AIST

ESIP Community

BESSIG

Recommended Practices

Education

Evaluation

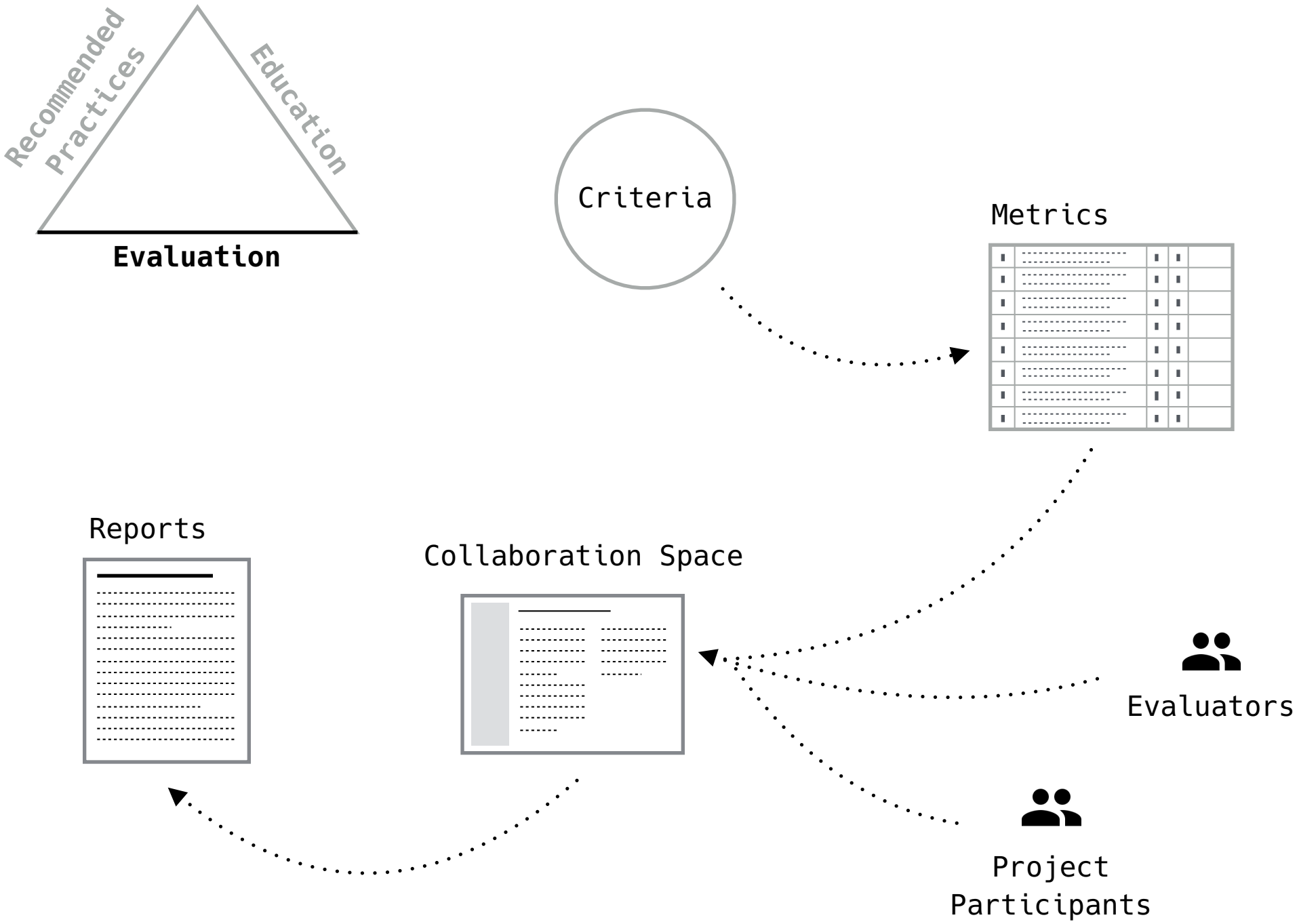
Software Carpentry?

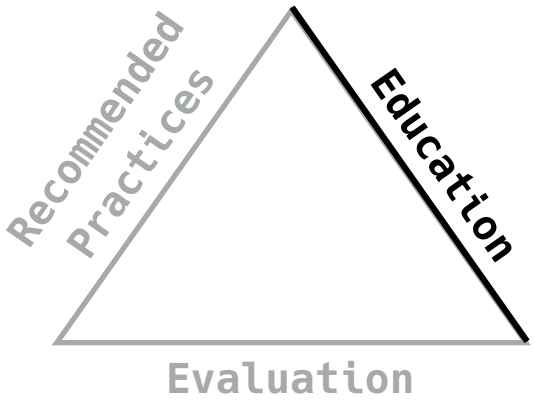
ESIP Community

NASA AIST @ ESIP

P&S Testbed

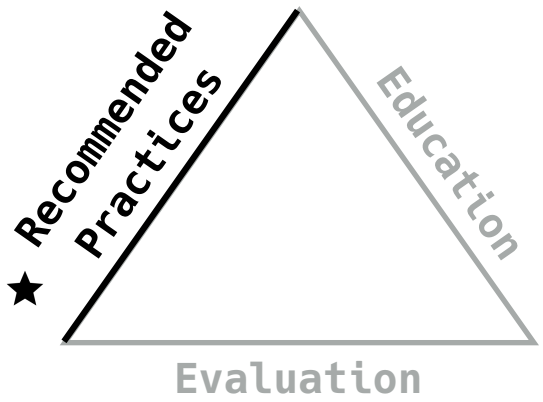
SemTech Testbed?


Recommended Practices
Education
Evaluation
Criteria
Metrics
Reports
Collaboration Space
Evaluators
Project Participants

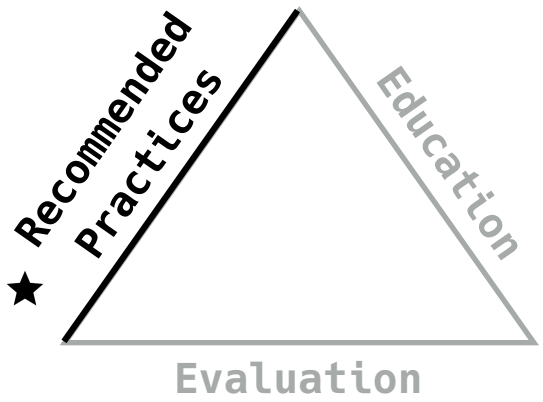


This boils down to building tools to support the culture we want to see. We want to evaluate and also to guide.

It is, for this conversation, something to keep in mind as we discuss criteria and the readiness levels/progressions developed based on those.

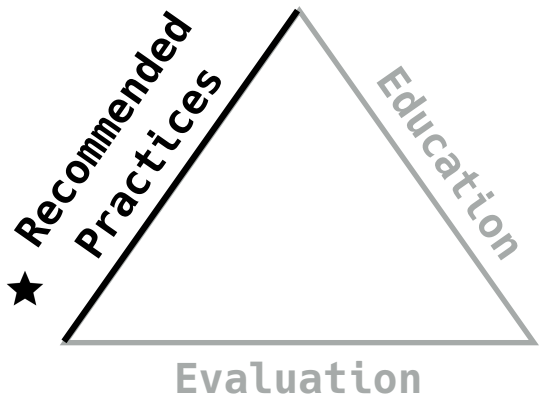


I’m assuming that we have noodled around the github site, looked at source materials?

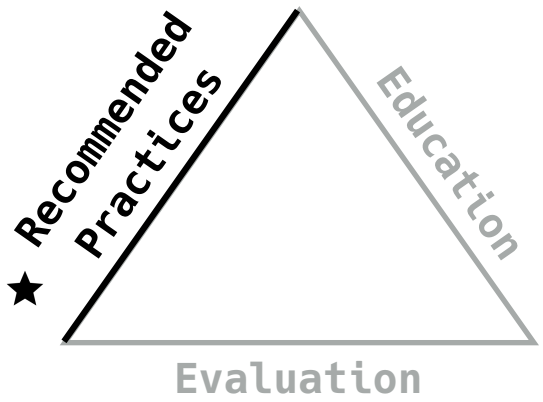


I am about to put my thumb on the scales.

We’re going to consider the criteria and metrics from a particular conceptual model to get at the use cases, like education, and to track with an ESIP implementation model.



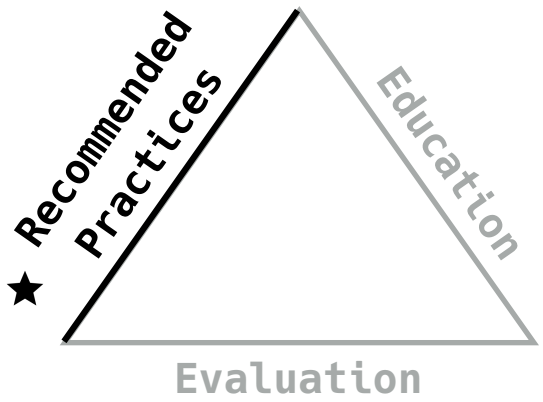
Criteria, Metrics and Progression



???

Good code practices

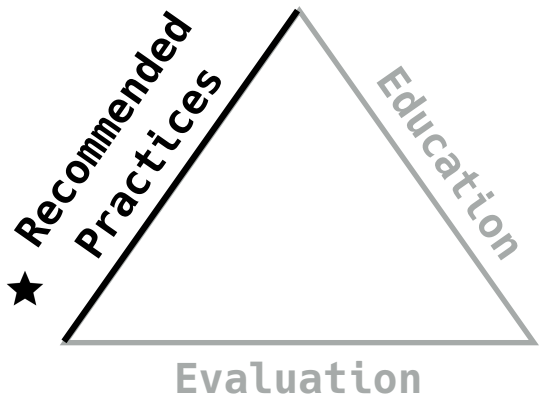
Software Progression
aka Readiness Levels



Currently

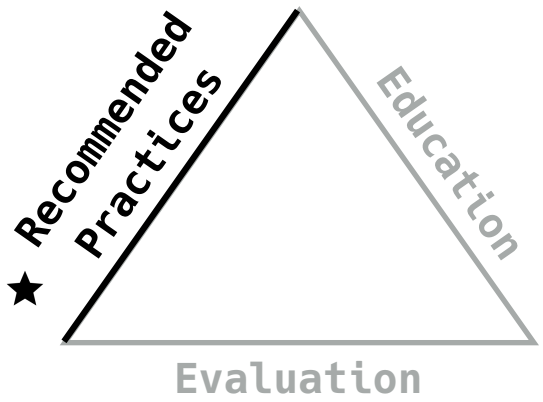
We have a bucket of criteria and apply some weighting scheme to get a metric.

And, in ESIP’s pilot AIST evaluation round, one of the outcomes was that the metric output matched what the evaluators felt was an appropriate TRL for the project.



Currently

But there was also a lot of uncertainty from the evaluators about specific questions and answerability. Utility. Applicability across kinds of software and technologies.



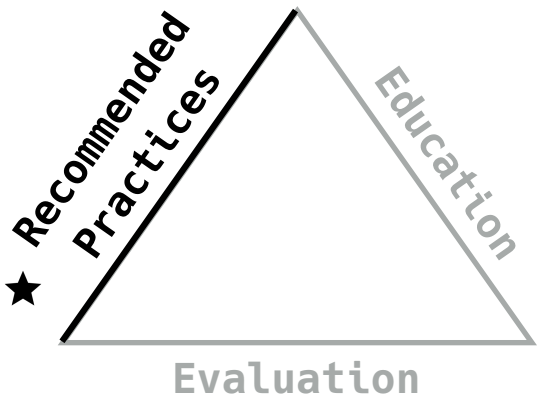
We’re changing the use case.

Education.

We want to make education a priority in this system.

So the components of the system need to make sense. The progression needs to make sense.

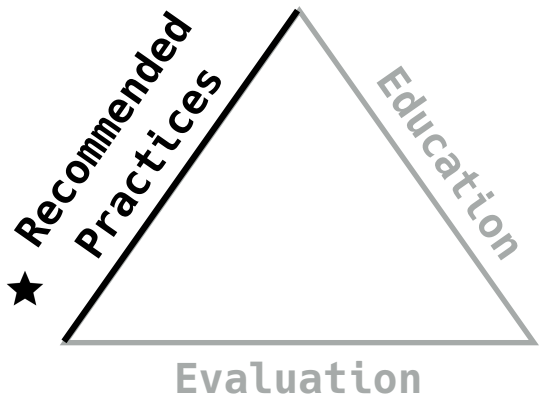
Scaffold instead of grab bag.



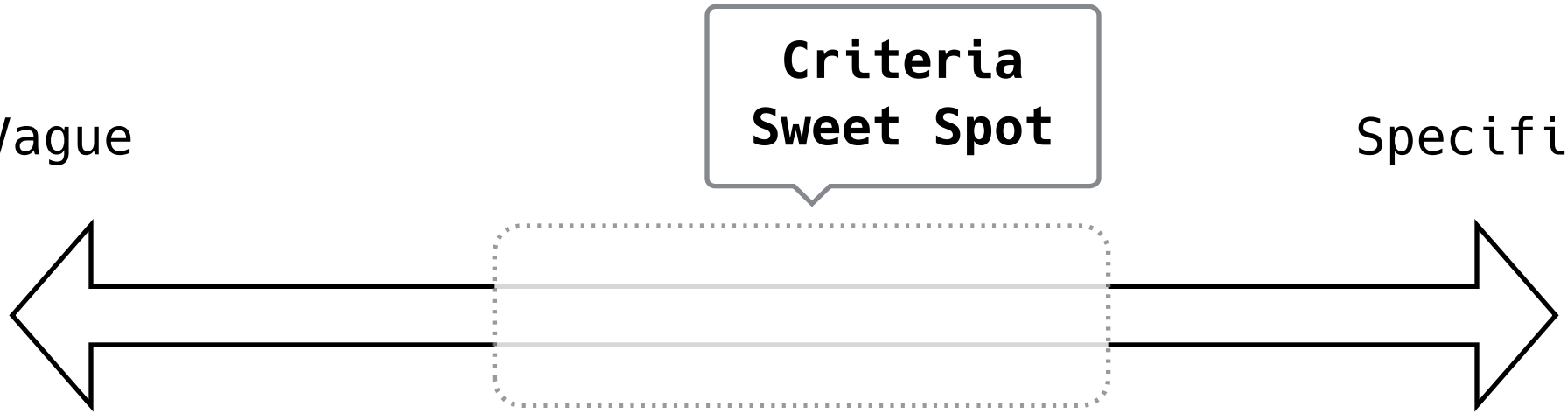
An Example

Continuous Integration

- Versioning
- Build System
- Provisioning



Levels of Abstraction

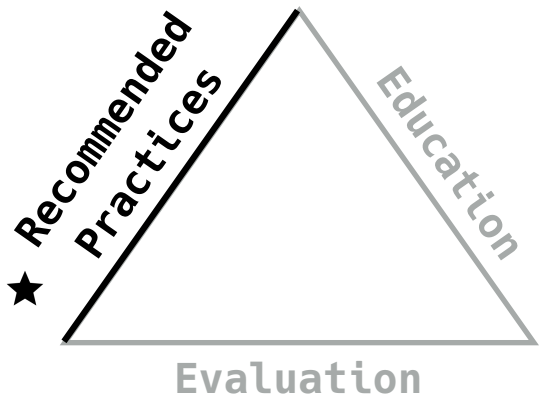


Not actionable

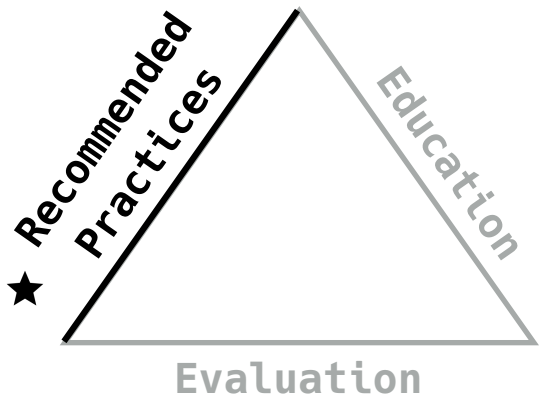
Hard to evaluate

Not widely applicable

Become dated quickly



Someone’s going to say Agile(TM)

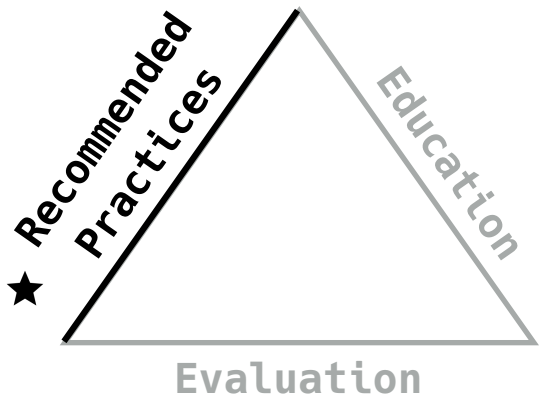


Ok. We've said it.

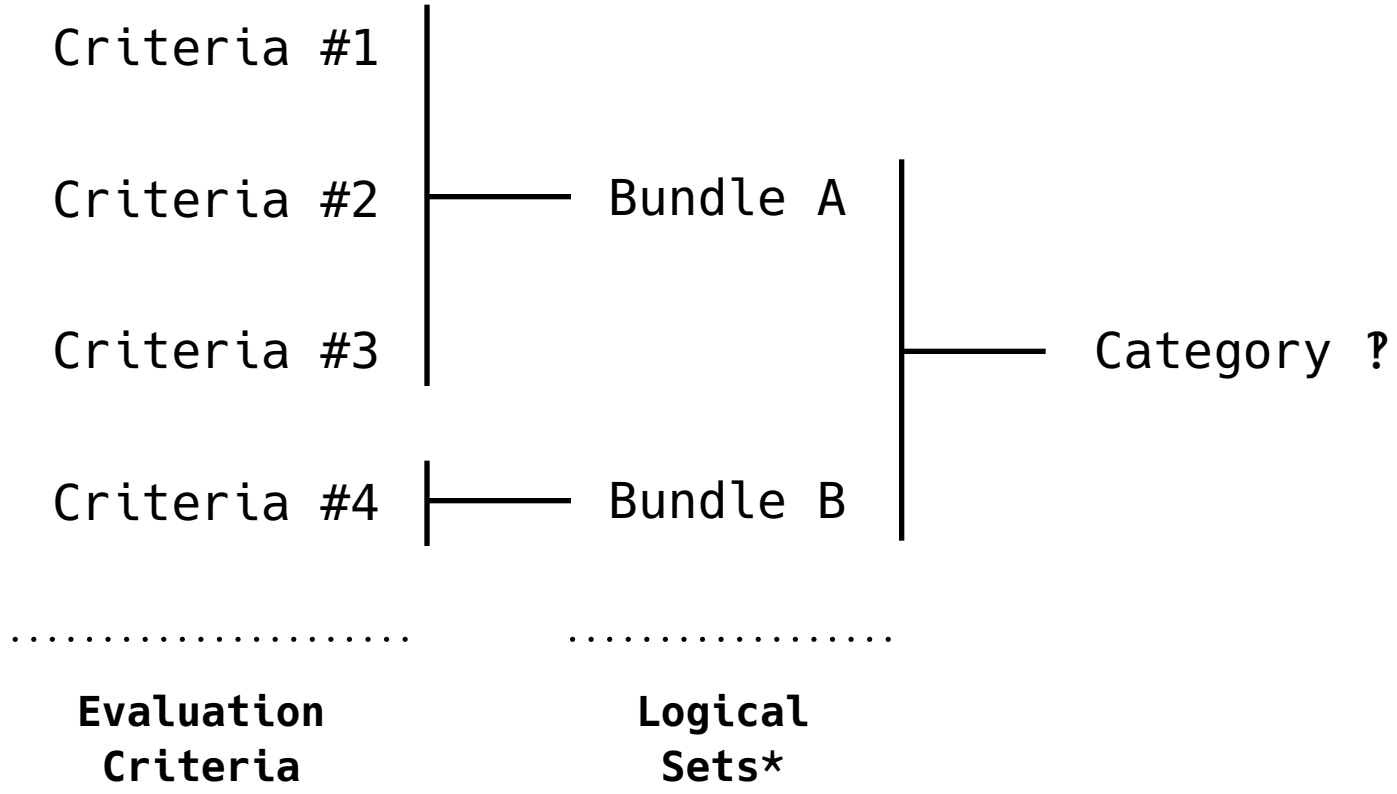
Agile(TM) is management.

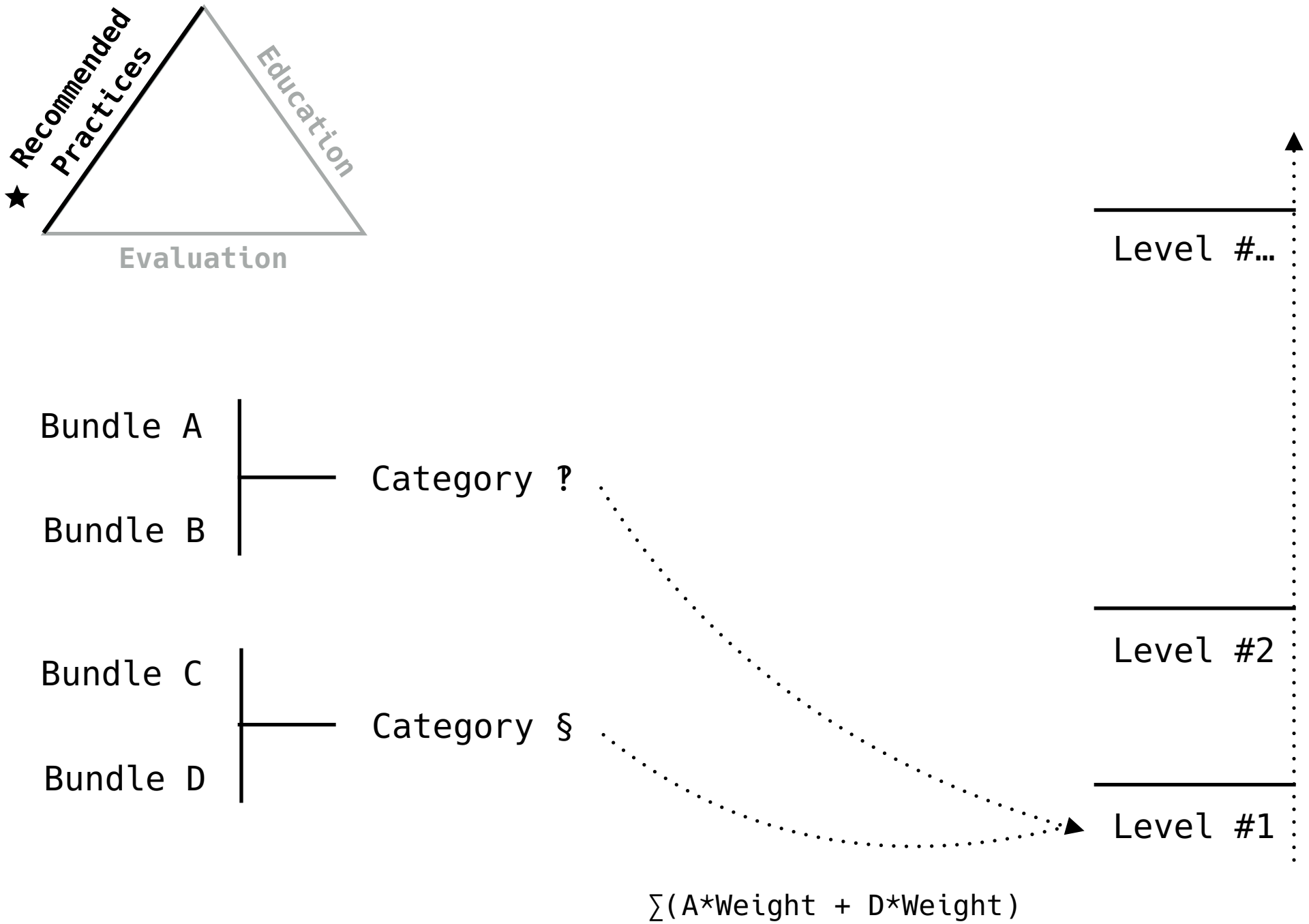
It's not magical good code pixie dust.

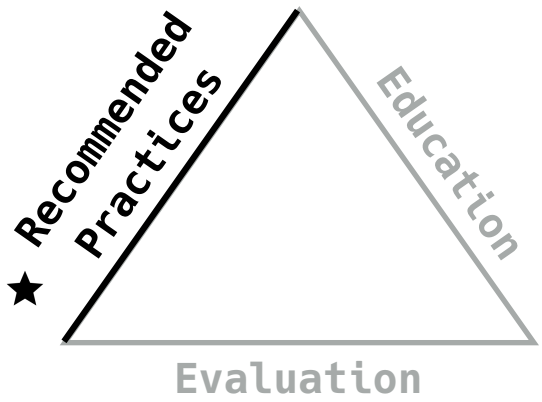
More importantly, it lacks evaluatibility in the kind of framework that is semi-automated or at least low impact.



A Conceptual Model




Recommended Practices
Education
Evaluation
Level #…
Bundle A
Category ?
Bundle B
Level #2
Bundle C
Category §
Bundle D
Level #1
∑(A*Weight + D*Weight)



Level #…

Bundle A

Bundle B

Category ?

Level #2

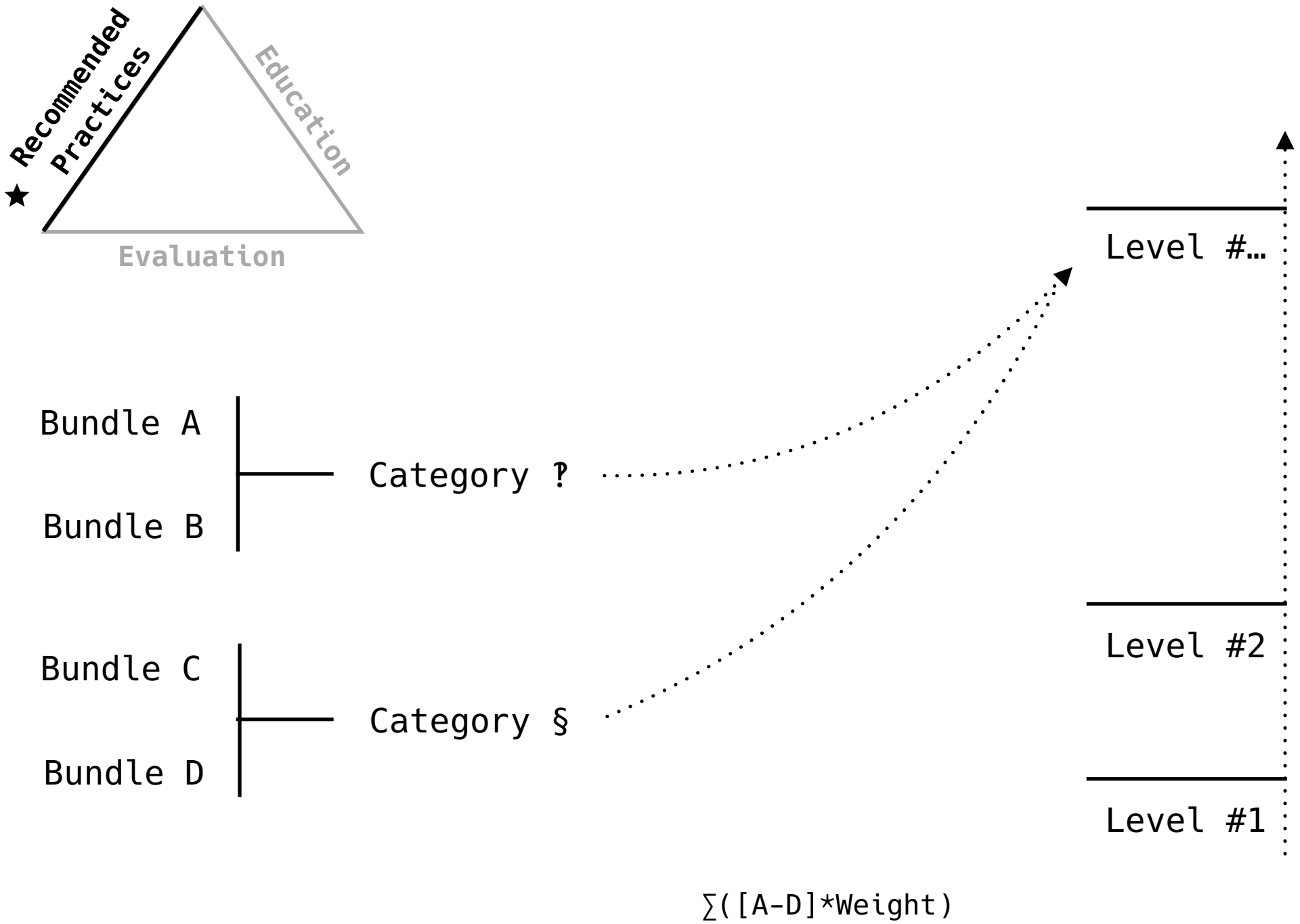
Bundle C

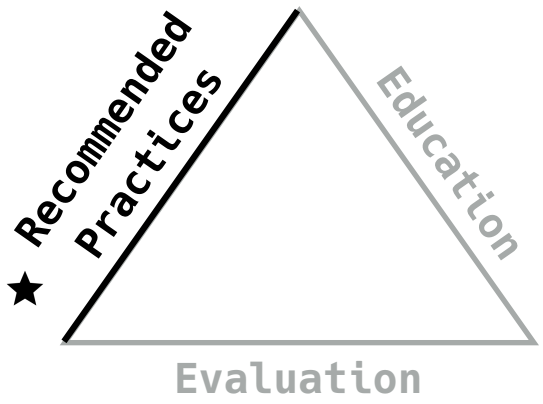
Bundle D

Category §

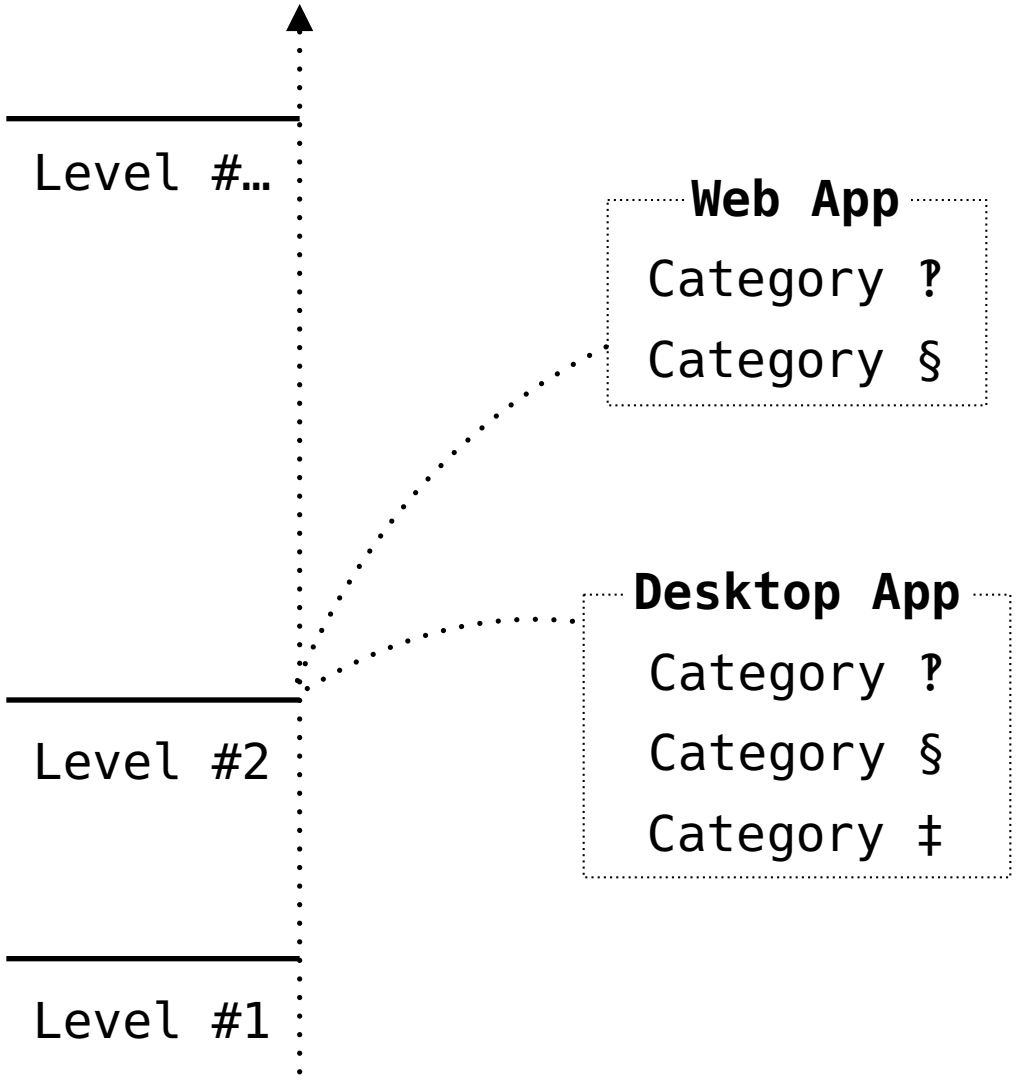
Level #1

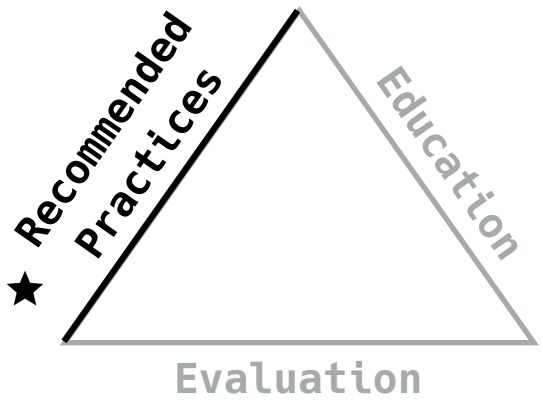
∑(A*Weight + B*Weight + D*Weight)


Recommended Practices
Education
Evaluation
Level #…
Bundle A
Category ?
Bundle B
Level #2
Bundle C
Category §
Bundle D
Level #1
∑([A-D]*Weight)



What types of software can we evaluate?



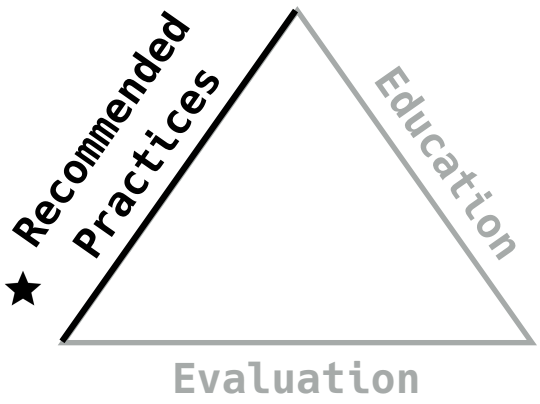


Another abstraction problem.

When we look at a piece of software like a plugin or an app built on top of an existing framework, how does that affect the criteria we use to evaluate it?

Are we evaluating the larger framework or the plugin?

What types of software can we evaluate?



Readiness || **Maintenance**

Level #…

Project Completed

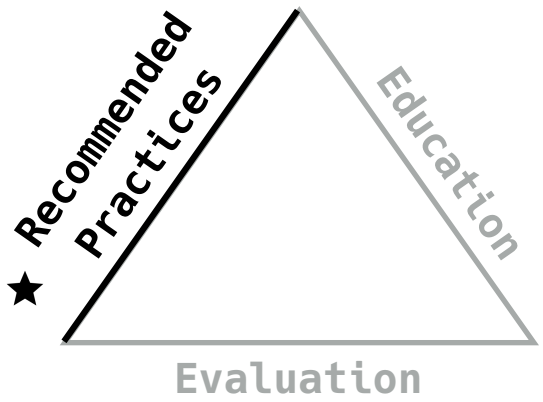
Research Object Governance & Publication

Long-term Maintenance

Level #2

Unmaintained Archive

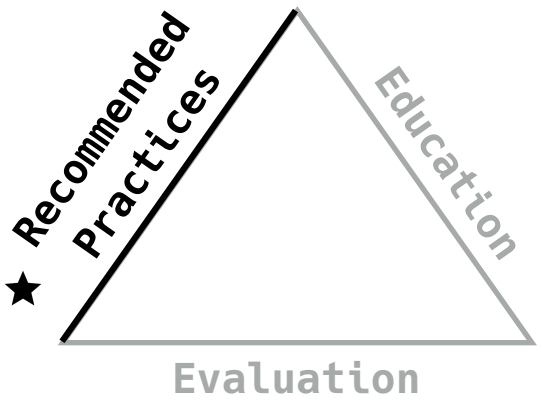
Level #1



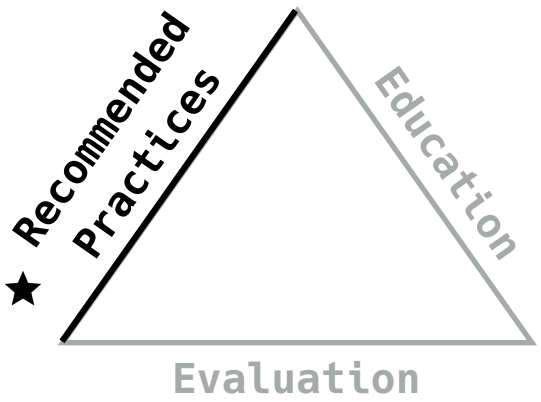
Reusability

How do we evaluate reusability?

Readiness Level + Maintenance + ?? = Reusable?



Our Approach



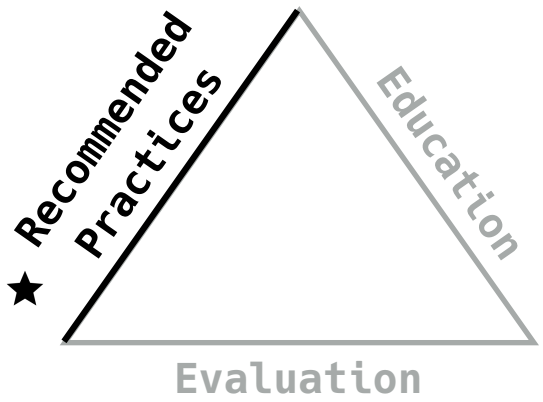
Planned Revision Activity

ESIP Guidelines

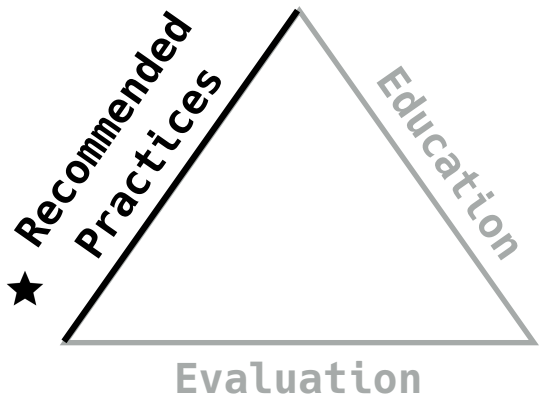
Category §

Category ⁇

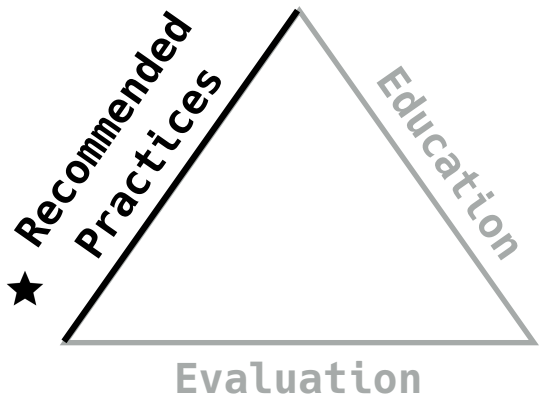
Category ‡



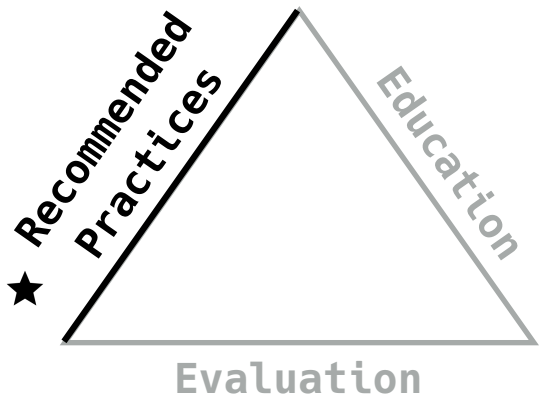
First, do the current categories capture the important concepts? What’s missing? What’s extra?



Second, do the criteria currently defined for each concept still apply? Are they out of date? Mismatched level of abstraction? Are we missing criteria?



Third, can we define readiness levels based on the concepts? Can we define education levels?



Finally, if you know of a dev that might be interested in a bit of meta-thinking, pass along the invite to participate.

To participate in the software evaluation evaluation process:

Resources:

http://roomthily.github.io/technology-evaluation-research/

Send me an email.

Add your email to the sign-in sheet.