Software engineering day-to-day priorities survey

Imagine that a software engineer has received code developed by a scientist, which ostensibly works correctly; what should be their top 3 priorities in how they spend their time (or, for (f), *don't* spend their time)? 1 = top priority, 2 = second, 3 = third.

(a) Modifiability, Readability & Maintainability

(b) Reusability (ability to share components, parameterizations or infrastructure pieces between modeling systems)

(c) Ease of use (smooth user interface, good error messages, documentation, user support, ease of porting)

(d) Performance & Scalability

(e) Assuring correctness through testing & code reviews, and designing for testability

(f) SEs should spend less time getting new developments into the model, even if it means sacrificing some of these other areas

(g) Training scientists to do these

Software engineering day-to-day priorities survey: results

(a) Modifiability, Readability & Maintainability: 15

- 1st: 4, 2nd: 6, 3rd: 5

(b) Reusability (ability to share components, parameterizations or infrastructure pieces between modeling systems): 10

- 1st: 1, 2nd: 4, 3rd: 5

(c) Ease of use (smooth user interface, good error messages, documentation, user support, ease of porting): 16

- 1st: 5, 2nd: 4, 3rd: 7

(d) Performance & Scalability: 17

- 1st: 5, 2nd: 7, 3rd: 5

(e) Assuring correctness through testing & code reviews, and designing for testability: 22

- 1st: 15, 2nd: 6, 3rd: 1

(f) SEs should spend less time getting new developments into the model, even if it means sacrificing some of these other areas: 1

- 3rd: 1

(g) Training scientists to do these: 3

- 3rd: 3

Possible discussion topics

- How to better encourage collaboration & contribution from outside
- (How) should we split up CIME?
  - Possibility of manage_externals for CIME's externals
- What limits should we place on adding CIME dependencies on non-standard python modules?
- New requirements for end-to-end workflows
- More general initialization capabilities (seasonal / decadal prediction)
- Generating CMIP-compliant data out-of-the-box

CESM niche

What areas should software engineers focus on to differentiate CESM from other earth system models?

(Or is our strength our breadth?)