Effects of Enhanced Compiler Error Messages in Rust: A Preliminary Study

Computer Sciences
SCHOOL OF COMPUTER, DATA & INFORMATION SCIENCES
UNIVERSITY OF WISCONSIN-MADISON

Ziyi Zhang[1], Aiping Xiong[2]
University of Wisconsin-Madison[1], The Pennsylvania State University[2]

PennState
College of Information Sciences and Technology

Rust

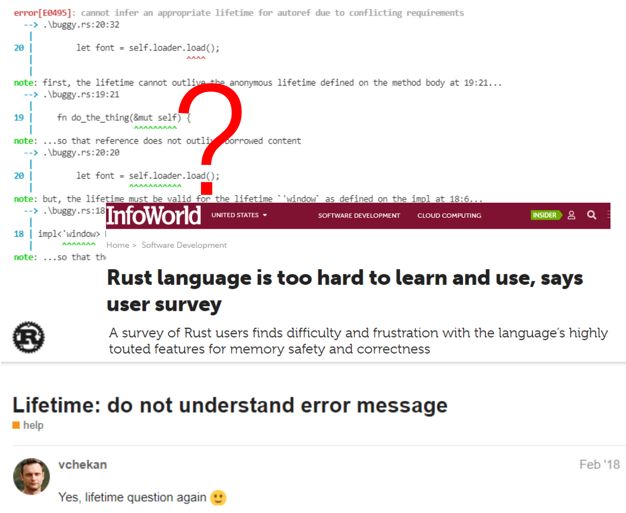
· A young systems programming language
- Providing high performance similar to C/C++
- Ensuring thread & memory safety

· Increasingly popular
- Most beloved language in the last six years
- Advocated by many big companies
- Adopted in many important projects

Difficulty of Using Rust

· Complex safety rules and strict compile-time check
- Ownership & lifetime

· Error messages can be confusing



```
error[E0495]: cannot infer an appropriate lifetime for autoref due to conflicting requirements
  --> .\buggy.rs:20:32
   |
20 |         let font = self.loader.load();
   |                                ^^^^
   |
note: first, the lifetime cannot outlive the anonymous lifetime defined on the method body at 19:21...
  --> .\buggy.rs:19:21
   |
19 |     fn do_the_thing(&mut self) {
   |                     ^^^^^^^^^
note: ...so that reference does not outli[ve] borrowed content
  --> .\buggy.rs:20:20
   |
20 |         let font = self.loader.load();
   |                    ^^^^^^^^^^^
note: but, the lifetime must be valid for the lifetime `'window` as defined on the impl at 18:6...
  --> .\buggy.rs:18
   |
18 | impl<'window>
   |      ^^^^^^^
note: ...so that th
```

InfoWorld UNITED STATES SOFTWARE DEVELOPMENT CLOUD COMPUTING INSIDER

Home > Software Development

Rust language is too hard to learn and use, says user survey

A survey of Rust users finds difficulty and frustration with the language's highly touted features for memory safety and correctness

Lifetime: do not understand error message

help

vchekan Feb '18

Yes, lifetime question again 🙂

Online User Survey

· Goal: improve the effectiveness of Rust error messages in understanding the error

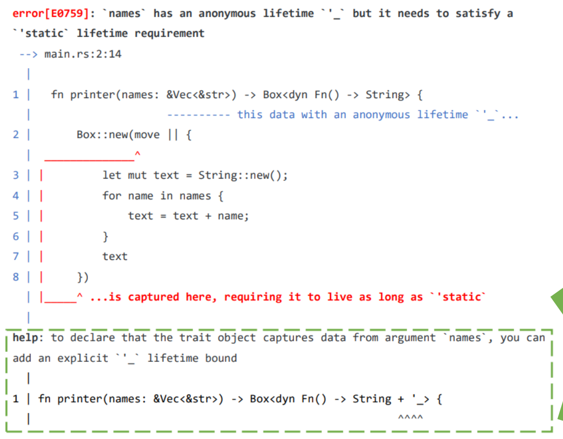
· Recruited Rust developers (N=52) from Rust forums (e.g., Rust User Forum)

· Participants were shown a Rust code snippet with varied error messages

· A 2✕3 mixed design
- One between-subject factor: enhanced type (solution, explanation)
- One within-subject factor: error message (w/o, original, an enhanced type)

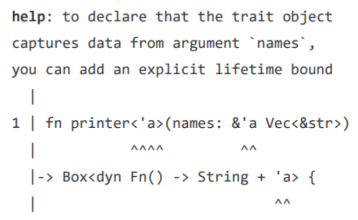
· Participants evaluated the error messages
- Difficulty of root cause identification
- Workload to comprehend using NASA TLX (Hart & Staveland, 1988)

Stimuli



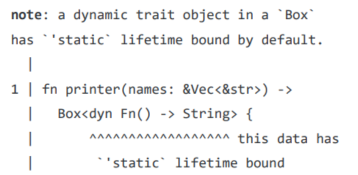
```
error[E0759]: `names` has an anonymous lifetime `'_` but it needs to satisfy a
`'static` lifetime requirement
 --> main.rs:2:14
  |
1 |  fn printer(names: &Vec<&str>) -> Box<dyn Fn() -> String> {
  |                    ---------- this data with an anonymous lifetime `'_`...
2 |     Box::new(move || {
  |  ______________^
3 | |       let mut text = String::new();
4 | |       for name in names {
5 | |           text = text + name;
6 | |       }
7 | |       text
8 | |   })
  | |_____^ ...is captured here, requiring it to live as long as `'static`
  |
help: to declare that the trait object captures data from argument `names`, you can
add an explicit `'_` lifetime bound
  |
1 | fn printer(names: &Vec<&str>) -> Box<dyn Fn() -> String + '_> {
  |                                                         ^^^^
```

Original error message



```
help: to declare that the trait object
captures data from argument `names`,
you can add an explicit lifetime bound
  |
1 | fn printer<'a>(names: &'a Vec<&str>)
  |           ^^^^         ^^
  |-> Box<dyn Fn() -> String + 'a> {
  |                            ^^
```

Enhanced solution



```
note: a dynamic trait object in a `Box`
has `'static` lifetime bound by default.
  |
1 | fn printer(names: &Vec<&str>) ->
  |   Box<dyn Fn() -> String> {
  |       ^^^^^^^^^^^^^^^^^^ this data has
  |        `'static` lifetime bound
```

Enhanced explanation

References
Sandra G.Hart, Lowell E.Staveland. 1988. Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research. In P. A. Hancock & N. Meshkati (Eds.), *Human mental workload* (pp. 139–183). North-Holland.
Shuofei Zhu, Ziyi Zhang, Boqin Qin, Aiping Xiong, Linhai Song. 2022. Learning and Programming Challenges of Rust: A Mixed-Methods Study. Accepted in ICSE 2022

Results

Difficulty rating

· 3 (error message: w/o, original, enhanced)×2 (enhanced type: explanation, solution) mixed ANOVA
- Only the main effect of error messages was significant
- Post-hoc analysis: task with enhanced messages was rated easier than the original one

Mean values of difficulty rating answers

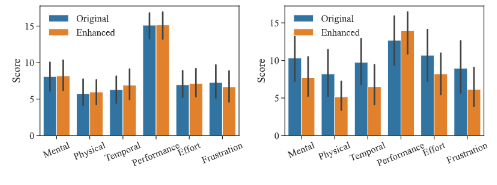
	w/o	Original	Enhanced
Explanation (33)	6.00	4.55	4.19
Solution (19)	7.05	5.32	4.21

NASA TLX (6 subscales)

· The main effect of error message was significant

· The two-way interaction of enhanced type and error message was significant
- Enhancement by solution was more evident than that by explanation

· The main effect of subscale was significant
- Higher ratings on performance



(a) Explanation (b) Solution

Conclusion

· Enhanced messages improve users' understanding

· Enhanced solution is more effective than enhanced explanation

Future work: recruit more participants for more balanced data