Contribution ID: **319**

Type: **not specified**

Rust language evolutions for better kernel developer experience

This activity comprises of two parts. First it will be a short update on the development of language features as we close the chapter on the Rust project goal 2025H2, covering features like arbitrary_self_types and trait evolution and the projected availability of the features on stable Rust releases and focusing on how this enables better kernel developer experience when working on Linux kernel code.

Second, we will engage panel discussion on two important language features that we aim to bring into complete feature in the coming year of 2026. First is the in-place initialisation which is the continuation of in-place initialisation programme from the project goal 2025H2. In the panel discussion, we will first present briefly three proposals from both Rust-for-Linux team and other Rust language team members: "init" expression, out-pointer and guaranteed emplacement. In the panel discussion, we will collect feedback on the ergonomics of these three proposals with worked examples derived from existing in-tree Rust code; concerns on uncovered use cases and safety issues; as well as suggestions to the language designs and potentially new approaches.

If time permits, we would like to also have a discussion on trait evolution as the second part and how it would help with refactoring the kernel crate's trait hierarchy. We will first present worked example of this language feature using existing code in the kernel crate, and collect feedback on the ergonomics and use case from the kernel developers.

Primary author: DING, Xiangfei

Presenter: DING, Xiangfei

Session Classification: Rust MC

Track Classification: Rust MC