TSGL-Rust: Teaching Concurrency Through Compile Time Enforcement via Rust-Based Visualization of Classical Synchronization Problems

IIUM Engineering Journal • Article • *Open Access* • 2026 • DOI: 10.31436/iiumej.v27i2.4361

Gunawan, Teddy Surya

Electrical and Computer Engineering Dept., International Islamic University Malaysia, Malaysia

Abstract

Concurrent programming remains a persistent challenge in undergraduate computer science and engineering education, largely because its failure modes are difficult to observe and reason about. Race conditions and deadlocks often emerge only under specific thread interleavings, leaving students to reconcile correct-looking code with unpredictable runtime behavior. Prior instructional approaches, such as the Thread-Safe Graphics Library (TSGL), address this issue by enabling real-time visualization, but they rely on languages that do not enforce safe access to shared state. This separation between observation and enforcement leaves a gap between how concurrency errors are detected and how they are prevented. This paper presents TSGL-Rust, a Rust-based reimplementation of classical TSGL visualizations that integrates compile-time safety guarantees with runtime visualization. Implemented using the egui/eframe framework, the system reproduces

three canonical synchronization problems, namely Producer-Consumer, Reader-Writer, and Dining Philosophers, while shifting certain classes of errors from runtime to compilation through Rust's ownership model and borrow checker. The architecture deliberately separates concurrency logic from rendering through a crossbeam channel, decoupling worker threads from the egui main-thread render loop. This paper is presented as a system and tool description: it focuses on architectural design, the mapping from C + + TSGL primitives to Rust constructs, the channel-based communication pattern, and the instructional affordances that follow from these design choices. The contribution lies in the system itself and in the boundary it makes visible between data-race prevention and logical errors, such as deadlock, rather than in any claim about measured learning gains.

Author keywords

compile-time safety; Concurrent programming; operating systems education; Rust; visualization

Funding details

Details about financial support for research, including funding sources and grant numbers as provided in academic publications.

Funding sponsor	Funding number	Acronym
International Islamic University Malaysia See opportunities by IIUM ↗		IIUM
TSGL-Rust		

Funding text

This work is partially supported by and also conceptually aligned with ongoing research under the AOARD project FA2386-25-1-4061 and SPI25-295-0295, \u201CAdaptive Physics Guided Transformer with Explainable Attention for Atmospheric Turbulence Mitigation,\u201D particularly in relation to concurrent data acquisition, inference, and visualization pipelines. The author thanks the students of the EECE 4342 Operating Systems course at IIUM for their engagement during the development and classroom deployment of TSGL-Rust. The author used generative AI tools (ChatGPT and Claude) solely for language refinement and structural editing during manuscript preparation; all AI-assisted text was reviewed and verified by the authors, and no AI was used to generate research data, results, or substantive technical content.